MICROMIKKO

TECHNICAL MANUAL

First edition August 1982

Our policy is one of continuous development and improvement; we reserve the right to alter technical data without notice.

Nokia Electronics assumes no responsibilities for any errors that may appear in this document.

Copyright C 1982 by NOKIA CORPORATION, Electronics,
Helsinki Finland

CONTENTS page

SECTION 1, HARDWARE OVERVIEW 1

Typical MicroMikko Configuration 3

SECTION 2. MicroMikko operation 4

Parts location in control unit 5

Control unit block diagram 6

Display unit block diagram 7

Keyboard block diagram 8

Controller board 9

Controller board data flow 11

2.1 Central processing unit 12

8085 Block diagram 13

8085 Instruction set summary 14

8085 Instruction set 15

Addressing logic 30

Dynamic RAM refresh 30

MicroMikko address space 31

8 bit data register 32

Interrupts 33

2.2 DMA Usage and Programming 34

DMA-channel definitions 34

DMA-controller block diagram 34

Functional description 35

DMA-operation 36

Idle cycle 36

Active cycle 37

Single transfer mode 37

Blcok transfer mode 37

Demand transfer mode 37

page

TRANSFER TYPES 37
Memory to memory 38
Auto initialize 38
Priority 39
Compressed timing 39
Address generation 39

REGISTER DESCRIPTION 40
Current address register 40
Current word register 40
Base address and base Word Count Register 40
Command Register 41
Mode Register 41
Request Register 41

REGISTER BIT LEVEL DEFINITIONS 42
Command Register 42
Mode Register 43
Request Register 43
Mask Register 44
Status Register 45

Temporary Register 45
Software Commands 46
Clear First/Last Flip-Flop 46
Master Clear 46
DMA COMMAND CODES 47

2.3 KEYBOARD INTERFACING 48
Keyboard scan codes
Finnish/Swedish 49
Norwegian/Danish 51
UK 53
Germany 55
US ASCII 57

page

2.4 CRT CONTROL 59
CRT controller initialization 59
DMA initialization for CRT purposes 59
Functional description of CRT-controller. 60
Block Diagram 60
Counters 61
Character counter 61
Line counter 61
Row counter 61
Raster timing and video controls 61
Row buffers 61
FIFO's 62
Buffer I/O Controllers 62
General Systems operational description 62
Display of a character ROW 62
Display Row Buffering 63
Display format 64
Screen format 64
Row Format 65
Raster Timing 67
DMA Timing 68
Interrupt Timing 69
VISUAL ATTRIBUTES AND SPECIAL CODES 69
Special Codes 69
Field Attributes 70
Cursor Timing 73
DEVICE PROGRAMMING 74
Instruction set 74
Reset Command 75
Start Display Command 78
Stop Display Command 79
Load Cursor Position 79
Enable Interrrupt Command 80

page

Disable Interrupt Command 80
Preset Counters Command 80
Status Flags 80
VALUES SET NORMALLY IN MICROMIKKO 82
MicroMikko Character sets 84

2.5 BAUD RATE GENERATOR 87
Line Speed Characteristics 87
Operational description 88
Normal initialization in MicroMikko 93

2.6 MPSC 94
General description 94
Functional description 94
Asynchronous operations 95
Asynchronous mode Register Set up 98
Synchronous operation - Monosyn, Bisync 98
Synchronous mode register set up - Monosync, Bisync 100
Synchronous operation - SDLC 103
Synchronous mode register setup SDCL/HDLC 104
Detailed command description 105
Detailed register description 107
WR0 107
WR1 109
WR2 channel A 111
WR2 channel B 113
WR3 114
WR4 115
WR5 117
WR6 119
WR7 119
RR0 120
RR1 124
RR2 126

page

System interface 126
Polled opeartion 127
Wait operation 128
Interrupt driven operation 128
Priority resolution: Vectored Mode 128
DMA operation 130
Basic initialization of MPSC in MicroMikko 131

2.7 Floppy Disk Control 133
Block diagram 133
8272 Register 134
Polling feature of 8272 137
8272 command set 138
Command descriptions 140
Read data 140
Write data 143
Write deteted data 144
Read deleted data 144
Read a track 145
Read ID 145
Format a track 145
Scan commands 146
Seek 148
Recalibrate 149
Sense interrupt status 149
Seek, Interrupt codes 151
Specify 151
Sense drive status 152
Invalid 152
Status registers 153

2.8 Interfaces/connectors 162
Standard cable list 165

2.9 Test & Diagnostics 166

page

3.0 SOFTWARE 168

CP/M operating system 168

Memory usage 169

File management 170

Diskette organization 172

Reserved locations in page zero 173

The BIOS entry points 174

IOBYTE 176

SYSTEM function summary 178

Line and Printer Channel Freq. change 179

Real time Clock 181

Keyboard code conversion 182

Menu & Inittask 184

Transmit & Receive 188

Terminal features 195

4.0 Notes of other documentation 205

5.0 BIOS listing 206

SECTION 1

HARDWARE OVERVIEW

The Micro-Mikko system has three major elements; a Control Unit, a Display unit and a keyboard. In addition a various types of printers can be attached to the Control unit.

The Control Unit is the heart of your Micro-Mikko system. The Control Unit houses the microprocessor, Read-Only-Memory, Read/Write Memory, Power Supply as well as interfaces for standard peripherals and a communication line. One or two 5-1/4" Diskette Drives can also be mounted in the Control Unit providing 160 kB or 320 kB storage each.

The Controller Board is a large board fitted horizontally in the base of the Control Unit and includes microprocessor, 4 kB ROM, 64 kB RAM and interface circuits to the Display unit, keyboard, Diskette drives, communication line and the peripheral line.

The Control Unit power system operates at 110, 120, 220, 240 V $\pm$ 10 %. Frequency is 50 hz $\pm$ 1 hz. Power consumption is 140 VA. Power outlet should be connected to earthsocket.

The keyboard is attached to the Control Unit with a light-weight cable. The slim lowprofile keyboard features 83 keys, and offers commonly used data and word processing functions in a design combining the familiar typewriter and calculator pad layouts.

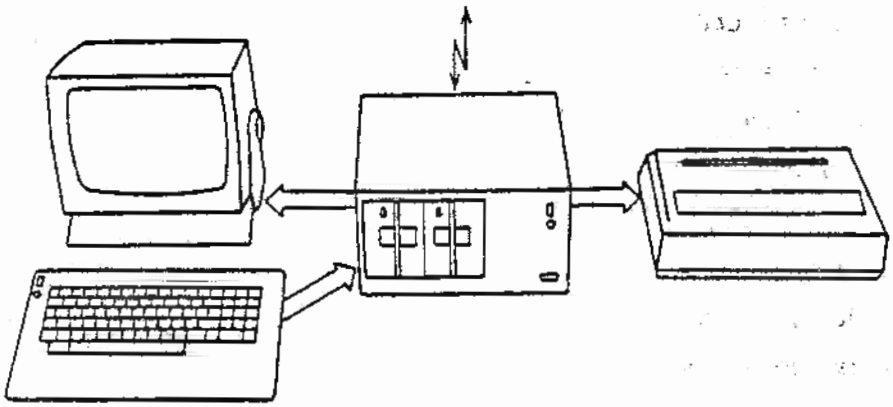
The Micro-Mikko Display unit is a high resolution green phosphor display offering the user quality usually found on larger and more expensive computer systems. The display features a 12" non reflecting screen. The screen displays 24 lines of 80 characters and an additional 25th line for system information. It supports 96 Scandinavian ASCII characters and in addition 16 special characters in Special Character Mode. Standard graphic is built up of 160*48 dots and this graphic can be used with special characters and normal character set. High resolution graphic supports 160*72 dots with limited character set. As an option variable types of other characters sets are also supported.

The Communications Interface provides a channel to data processing or input/output devices outside of your immediate system. These can be connected by using telephone lines and a plug-in modem, or directly by cable when the device is nearby.

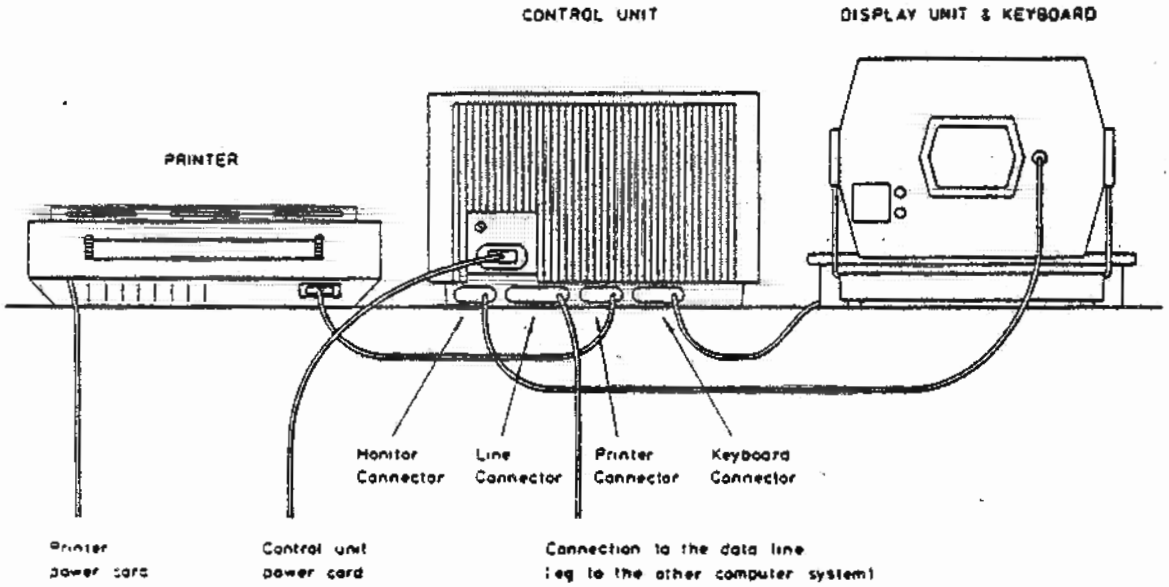
This facility utilizes the CCITT V,24/V,28 (EIA RS 232C) or optionally X.21. The line can be selected to be Asynchronous or Synchronous supporting transmission speeds, from 75 to 9600 BPS. One 25 pin "D" shell, male type connector is provided to attach various peripheral devices.

Basic printer version is of type 7*7 matrix. It supports 80 or 132 characters per line with speed of 100 characters per second at 80 characters per line. The printer uses asynchronous serial interface.

A wide range of other types of printers can also be used.



MicroMIKKO units



Rear view of MicroMIKKO showing cable connections

SECTION 2

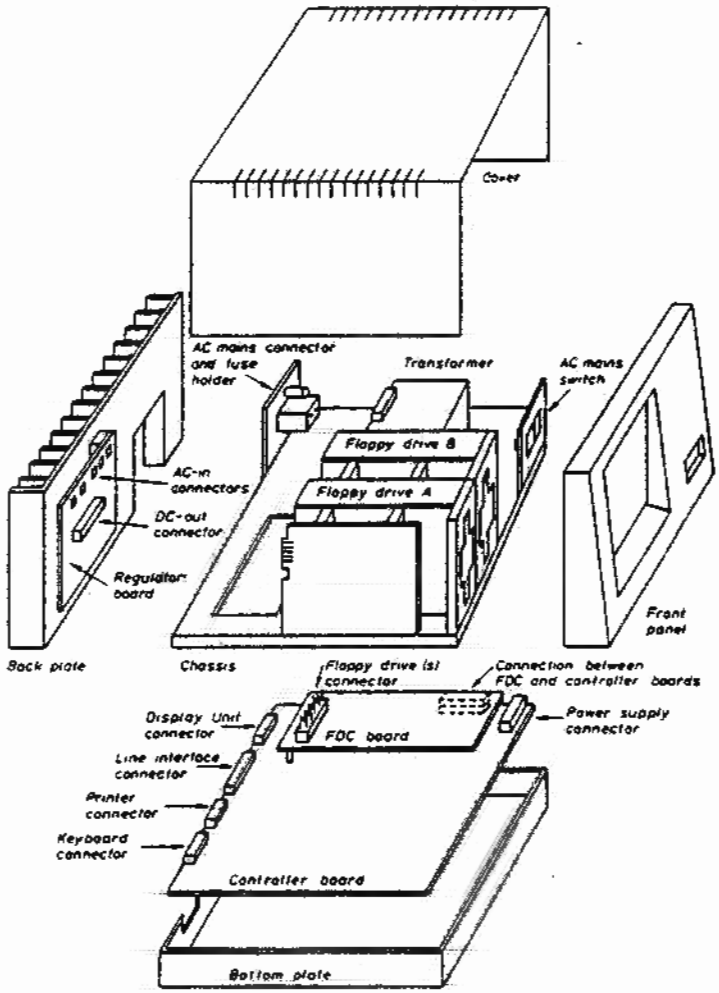
MICROMIKKO OPERATION

Parts location

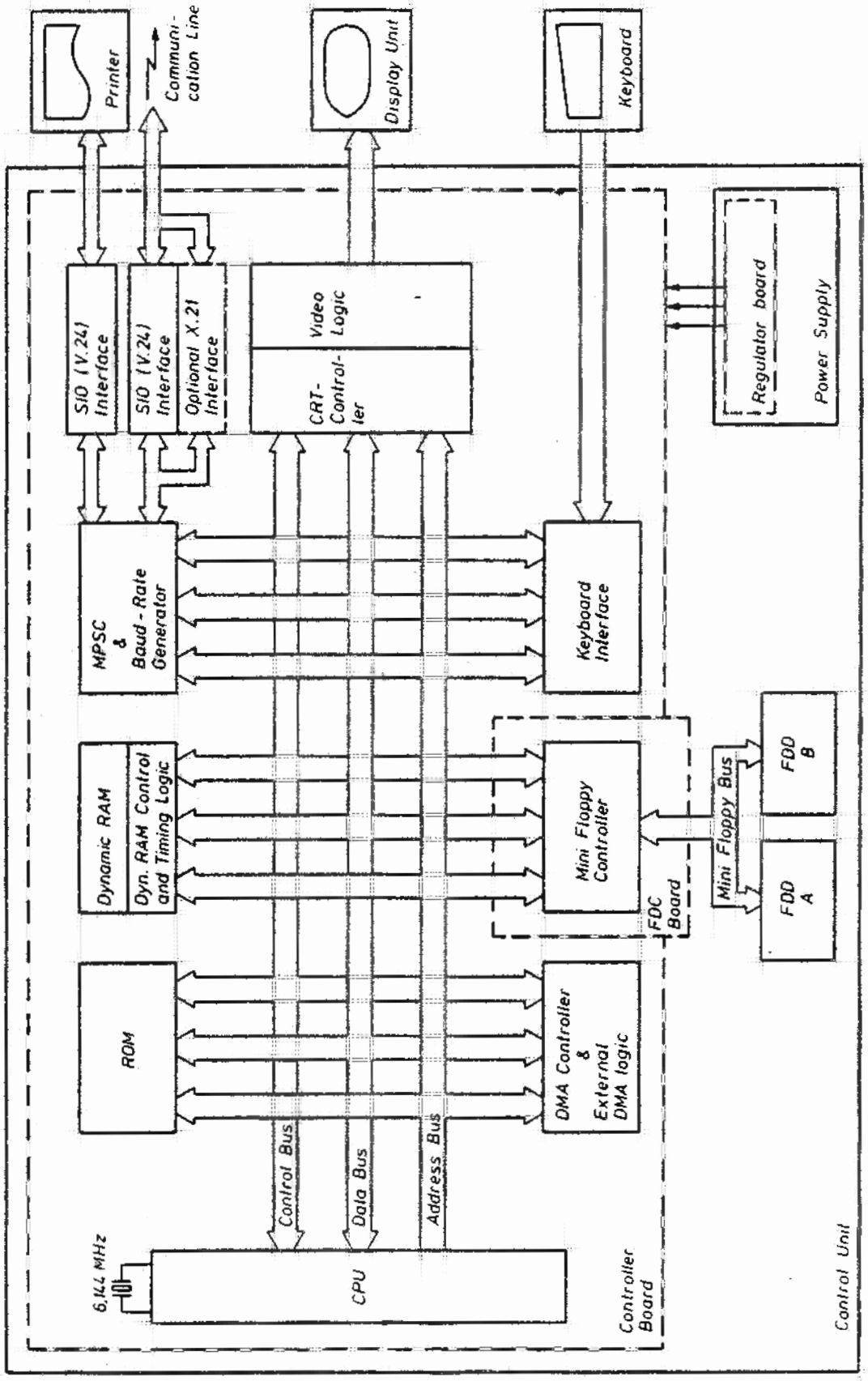
Controller board
Central Processing Unit
Addressing logic
8-Bit data register
Interrrupts
DMA
Keyboard interfacing
CRT control
Baud rate generator
MPSC
Floppy disk controller

PARTS LOCATION

Control unit houses System board, Power Supply and room for two 5 1/4" diskette drivers. Parts location inside of control unit is shown in figure below.

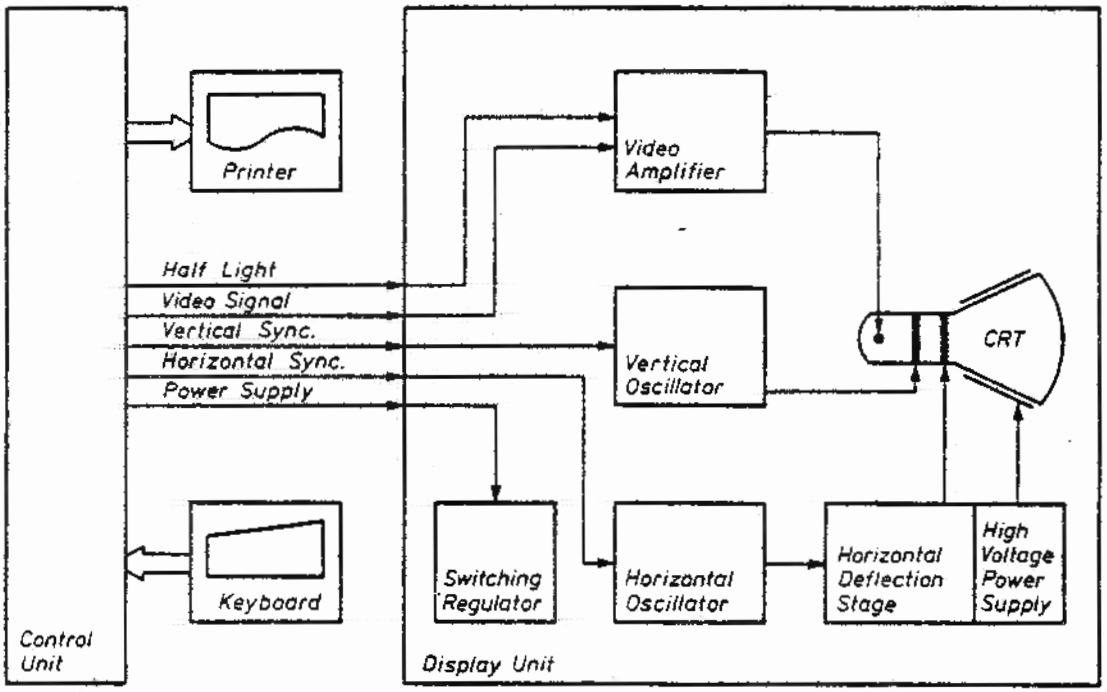


Parts location in Control Unit



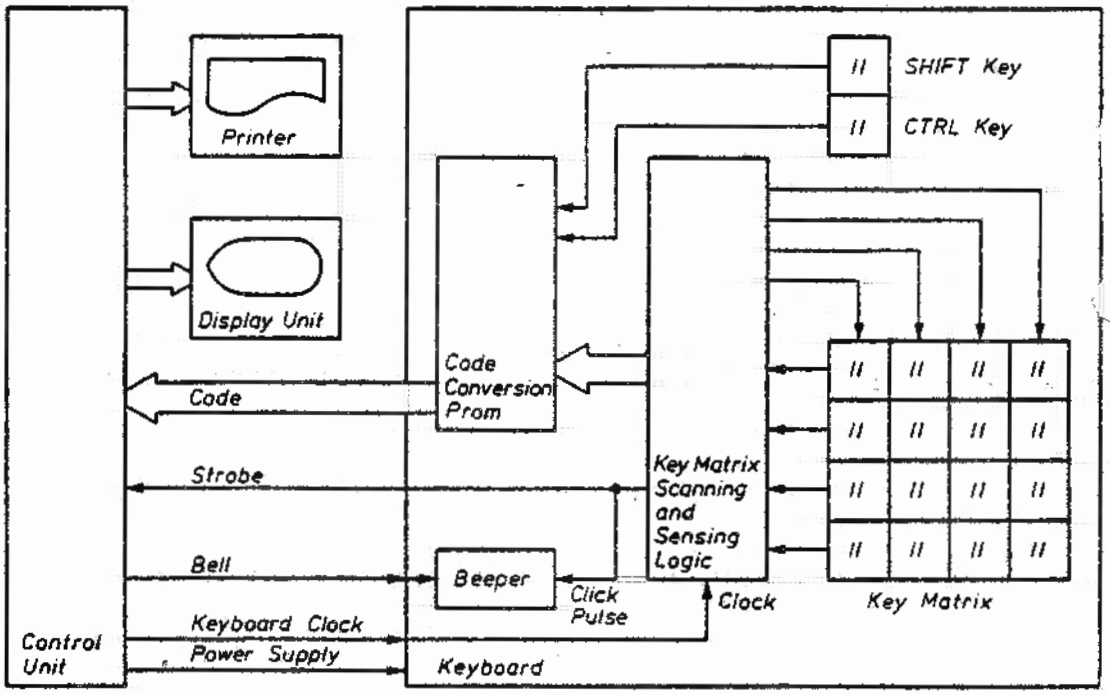
MicroMIKKO Control Unit Block Diagram

Display Unit houses a 12" green phosphor tube with magnetig deflection. All logic needed to provide deflection and beam intensity is located into the Display Unit.



Display unit.

Keyboard is of type slim and ergonomic. It features 83 keys located like a typewriter. All logic needed to provide multiplex operation from key to 8-bit parallel is located into the keyboard .



Keyboard

CONTROLLER BOARD

Controller board is a large board, fitted horizontally in the base of the Control Unit. Controller board is doublesided. DC-power enter the board through a connector. Other connectors for Main Channel and Help Channel are also located on the board. Jumpers on the board are factory preset.

The Major elements of the Controller board are divided into 9 major functional areas. They are, the processor subsystem and its support elements, the Read-Only-Memory (ROM) subsystem, the Read/Write Memory (RAM) subsystem, Direct Memory Access (DMA) subsystem, Keyboard interface logic, CRT control & videologic subsystem, Baud Rate Generator with Multi Protocol Serial Controller (MPSC) and Diskette drive interface.

The heart of the Controller board is the Intel 8085A microprocessor. This is an 8-bit microprocessor, its intruction set includes 80 assembler-language instructions, and its cycletime is 320 ns with a 6 Mhz quarz oscillator. 8085 houses 8-bit accumulator and 6 peaces of 8-bit registers, these can be used alone (8-bit) or coupled (16-bit). The processor supports 16 bits of addressing (64 kbytes of storage). Arithmetical operations are done in 8-bit Arithmetic Logical Unit (ALU). In addition to the internal dataregisters 8085 has 16-bit program counter (PC) and 16-bit Stack Pointer (SP). The operation of 8085 can be controlled with five external interrupt signals to support peripheral devices.

The Controller Board is designed to support both ROM and RAM. The Controller Board contains space for 4 K * 8-bit of EPROM. A module socket is provided, for this purpose. This EPROM contains Power-on Self-test and a bootstrap diskette loader. The EPROM is packaged in 24 pin module.

The Controller Board also contains 64 kbytes of Read/Write Memory. The memory is dynamic, 64K bits*8 chips.

The processor is supported by a set of high function support devices providing four channels of 16 bit Direct Memory Access (DMA) and three 16 bit timer counter channels combined with Multi Protocol Serial Controller (MPSC).

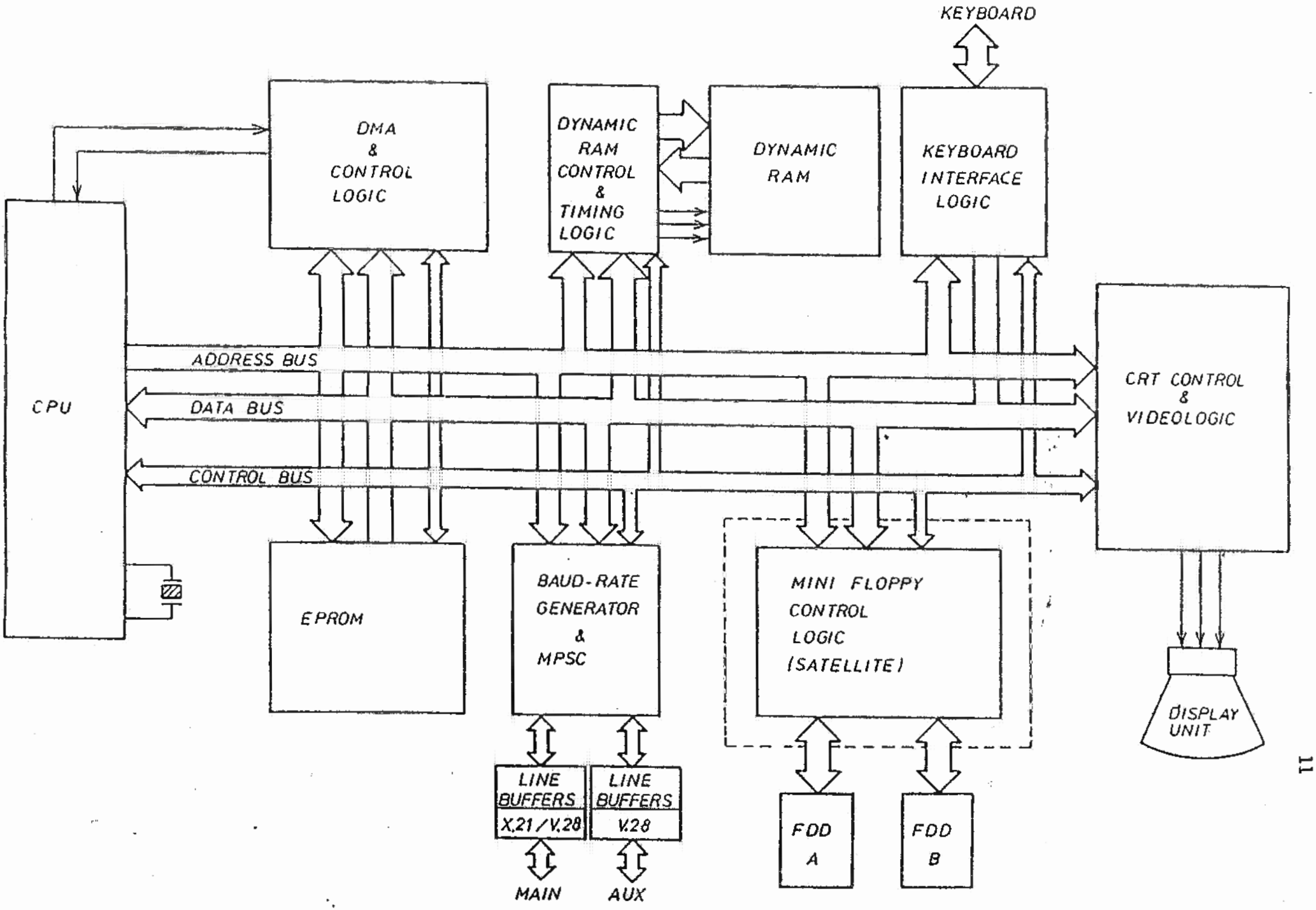
All the four DMA channels are available on the bus and are provided to support high speed data transters between I/O device and memory without processor intervention. DMA channel supporting CRT controller (channel 0) is also used to refresh the dynamic system memory. This is done along with the updating of display. This creates a memory read cycle with the intervals of 1,6 ms, which is short enough to refresh RAM.

Baud-rate generator, providing three timers/counters are used by the system to generate clock frequencies for MPSC. MPSC has the task to modify the parallel data used by the CPU to serial format and steer it to serial channel or vice versa. MPSC uses DMA and INT services when transferring data inside of the microcomputer.

The System board also contains the adapter circuits for attaching the 8-bit parallel interface from the keyboard. The keyboard interface generates an interrupt when a key is pressed, thus allowing the CPU to service the device.

CRT Controller and Videlogic contains CRT-controller with character-generator, DMA controllogic, shiftregister, parallel to serial, and timing logic.

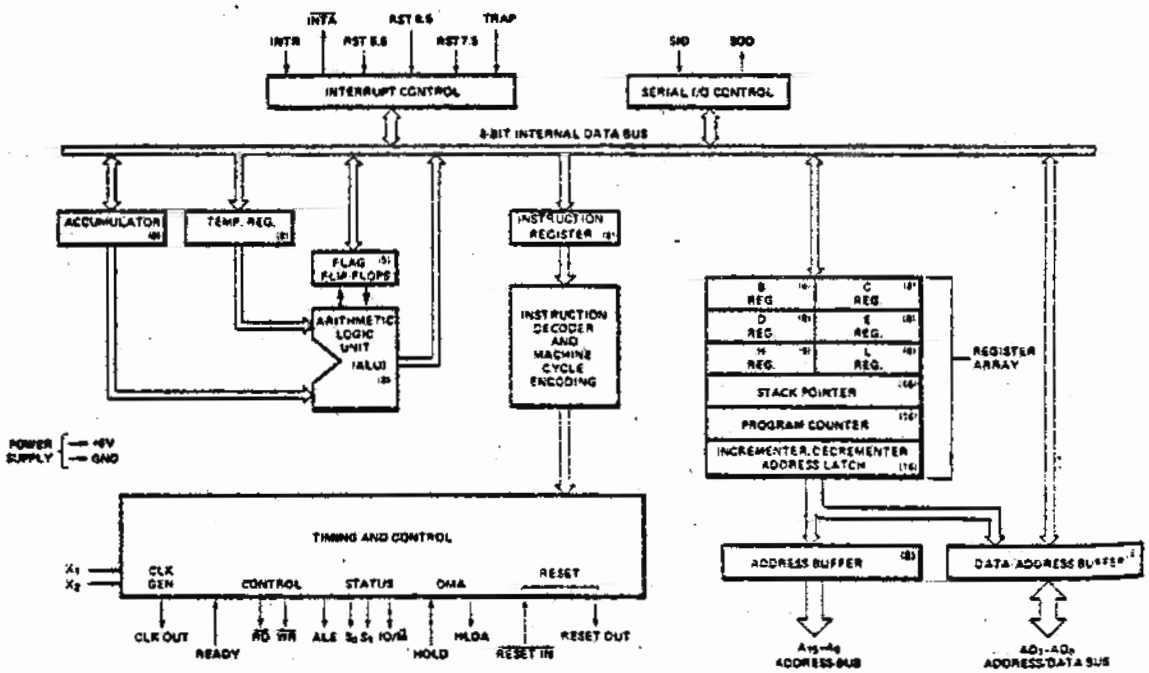
Diskette drive interface controls floppy disk drive operations.


KEYBOARD
DMA & CONTROL LOGIC
DYNAMIC RAM CONTROL & TIMING LOGIC
DYNAMIC RAM
KEYBOARD INTERFACE LOGIC
CPU
ADDRESS BUS
DATA BUS
CONTROL BUS
CRT CONTROL & VIDEOLOGIC
EPROM
BAUD-RATE GENERATOR & MPSC
MINI FLOPPY CONTROL LOGIC (SATELLITE)
DISPLAY UNIT
LINE BUFFERS X.21/V.28
LINE BUFFERS V.28
MAIN
AUX
FDD A
FDD B

2.1 CENTRAL PROCESSING UNIT

8085 houses an oscillator, that is steered with external crystal oscillator connected to CPU. Internal divider divides this 6,144 Mhz frequency by two, building system base clockfrequency. This is available in processor pin CLK OUT.

Into the RST pin of the CPU is connected a RC-circuit, that causes processor reset, when power is switched on. At the same time Program Counter and internal flipflops take their initial state. Peripheral circuits are set to their initial states with the signal RST OUT from CPU. Serial Output Data (SOD) pin is used to steer the beep-tone in the keyboard. After reset SOD is set to "0" and "beep" is activated. When the selftest finds that MicroMikko is not malfunctioning the beep-tone is turned off with setting SOD output to "1" . This tells also the User that MicroMikko is ready for use. Serial Input Data (SID) pin is used to read the status of Main Channel DSR (Data Set Ready).



Block Diagram of central Processing Unit

Instruction Set Summary

Mnemonic	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Operations Description
MOVE, LOAD, AND STORE									
MOV r1 r2	0	1	D	D	D	S	S	S	Move register to register
MOV M,r	0	1	1	1	0	S	S	S	Move register to memory
MOV r,M	0	1	D	D	D	1	1	0	Move memory to register
MVI r	0	0	D	D	D	1	1	0	Move immediate register
MVI M	0	0	1	1	0	1	1	0	Move immediate memory
LXI B	0	0	0	0	0	0	0	1	Load immediate register Pair B & C
LXI D	0	0	0	1	0	0	0	1	Load immediate register Pair D & E
LXI H	0	0	1	0	0	0	0	1	Load immediate register Pair H & L
STAX B	0	0	0	0	0	0	1	0	Store A indirect
STAX D	0	0	0	1	0	0	1	0	Store A indirect
LDAX B	0	0	0	0	1	0	1	0	Load A indirect
LDAX D	0	0	0	1	1	0	1	0	Load A indirect
STA	0	0	1	1	0	0	1	0	Store A direct
LDA	0	0	1	1	1	0	1	0	Load A direct
SHLD	0	0	1	0	0	0	1	0	Store H & L direct
LHLD	0	0	1	0	1	0	1	0	Load H & L direct
XCHG	1	1	1	0	1	0	1	1	Exchange D & E, H & L Registers
STACK OPS									
PUSH B	1	1	0	0	0	1	0	1	Push register Pair B & C on stack
PUSH D	1	1	0	1	0	1	0	1	Push register Pair D & E on stack
PUSH H	1	1	1	0	0	1	0	1	Push register Pair H & L on stack
PUSH PSW	1	1	1	1	0	1	0	1	Push A and Flags on stack
POP B	1	1	0	0	0	0	0	1	Pop register Pair B & C off stack
POP D	1	1	0	1	0	0	0	1	Pop register Pair D & E off stack
POP H	1	1	1	0	0	0	0	1	Pop register Pair H & L off stack
POP PSW	1	1	1	1	0	0	0	1	Pop A and Flags off stack
XTHL	1	1	1	0	0	0	1	1	Exchange top of stack, H & L
SPHL	1	1	1	1	1	0	0	1	H & L to stack pointer
LXI SP	0	0	1	1	0	0	0	1	Load immediate stack pointer
INX SP	0	0	1	1	0	0	1	1	Increment stack pointer
DCX SP	0	0	1	1	1	0	1	1	Decrement stack pointer
JUMP									
JMP	1	1	0	0	0	0	1	1	Jump unconditional
JC	1	1	0	1	1	0	1	0	Jump on carry
JNC	1	1	0	1	0	0	1	0	Jump on no carry
JZ	1	1	0	0	1	0	1	0	Jump on zero
JNZ	1	1	0	0	0	0	1	0	Jump on no zero
JP	1	1	1	1	0	0	1	0	Jump on positive
JM	1	1	1	1	1	0	1	0	Jump on minus
JPE	1	1	1	0	1	0	1	0	Jump on parity even
JPO	1	1	1	0	0	0	1	0	Jump on parity odd
PCHL	1	1	1	0	1	0	0	1	H & L to program counter
CALL									
CALL	1	1	0	0	1	1	0	1	Call unconditional
CC	1	1	0	1	1	1	0	0	Call on carry
CNC	1	1	0	1	0	1	0	0	Call on no carry
LOGICAL									
ANA r	1	0	1	0	0	S	S	S	And register with A
XRA r	1	0	1	0	1	S	S	S	Exclusive OR register with A
ORA r	1	0	1	1	0	S	S	S	OR register with A
CMP r	1	0	1	1	1	S	S	S	Compare register with A
ANA M	1	0	1	0	0	1	1	0	And memory with A
XRA M	1	0	1	0	1	1	1	0	Exclusive OR memory with A
ORA M	1	0	1	1	0	1	1	0	OR memory with A
CMP M	1	0	1	1	1	1	1	0	Compare memory with A
ANI	1	1	1	0	0	1	1	0	And immediate with A
XRI	1	1	1	0	1	1	1	0	Exclusive OR immediate with A
ORI	1	1	1	1	0	1	1	0	OR immediate with A
CPI	1	1	1	1	1	1	1	0	Compare immediate with A
ROTATE									
RLC	0	0	0	0	0	1	1	1	Rotate A left
RRC	0	0	0	0	1	1	1	1	Rotate A right
RAL	0	0	0	1	0	1	1	1	Rotate A left through carry
RAR	0	0	0	1	1	1	1	1	Rotate A right through carry

Mnemonic	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Operations Description
CZ	1	1	0	0	1	1	0	0	Call on zero
CNZ	1	1	0	0	0	1	0	0	Call on no zero
CP	1	1	1	1	0	1	0	0	Call on positive
CM	1	1	1	1	1	1	0	0	Call on minus
CPE	1	1	1	0	1	1	0	0	Call on parity even
CPO	1	1	1	0	0	1	0	0	Call on parity odd
RETURN									
RET	1	1	0	0	1	0	0	1	Return
RC	1	1	0	1	1	0	0	0	Return on carry
RNC	1	1	0	1	0	0	0	0	Return on no carry
RZ	1	1	0	0	1	0	0	0	Return on zero
RNZ	1	1	0	0	0	0	0	0	Return on no zero
RP	1	1	1	1	0	0	0	0	Return on positive
RM	1	1	1	1	1	0	0	0	Return on minus
RPE	1	1	1	0	1	0	0	0	Return on parity even
RPO	1	1	1	0	0	0	0	0	Return on parity odd
RESTART									
RST	1	1	A	A	A	1	1	1	Restart
INPUT/OUTPUT									
IN	1	1	0	1	1	0	1	1	Input
OUT	1	1	0	1	0	0	1	1	Output
INCREMENT AND DECREMENT									
INR r	0	0	D	D	D	1	0	0	Increment register
DCR r	0	0	D	D	D	1	0	1	Decrement register
INR M	0	0	1	1	0	1	0	0	Increment memory
DCR M	0	0	1	1	0	1	0	1	Decrement memory
INX B	0	0	0	0	0	0	1	1	Increment B & C registers
INX D	0	0	0	1	0	0	1	1	Increment D & E registers
INX H	0	0	1	0	0	0	1	1	Increment H & L registers
DCX B	0	0	0	0	1	0	1	1	Decrement B & C
DCX D	0	0	0	1	1	0	1	1	Decrement D & E
DCX H	0	0	1	0	1	0	1	1	Decrement H & L
ADD									
ADD r	1	0	0	0	0	S	S	S	Add register to A
ADC r	1	0	0	0	1	S	S	S	Add register to A with carry
ADD M	1	0	0	0	0	1	1	0	Add memory to A
ADC M	1	0	0	0	1	1	1	0	Add memory to A with carry
ADI	1	1	0	0	0	1	1	0	Add immediate to A
ACI	1	1	0	0	1	1	1	0	Add immediate to A with carry
DAD B	0	0	0	0	1	0	0	1	Add B & C to H & L
DAD D	0	0	0	1	1	0	0	1	Add D & E to H & L
DAD H	0	0	1	0	1	0	0	1	Add H & L to H & L
DAD SP	0	0	1	1	1	0	0	1	Add stack pointer to H & L
SUBTRACT									
SUB r	1	0	0	1	0	S	S	S	Subtract register from A
SBB r	1	0	0	1	1	S	S	S	Subtract register from A with borrow
SUB M	1	0	0	1	0	1	1	0	Subtract memory from A
SBB M	1	0	0	1	1	1	1	0	Subtract memory from A with borrow
SUI	1	1	0	1	0	1	1	0	Subtract immediate from A
SBI	1	1	0	1	1	1	1	0	Subtract immediate from A with borrow
SPECIALS									
CMA	0	0	1	0	1	1	1	1	Complement A
STC	0	0	1	1	0	1	1	1	Set carry
CMC	0	0	1	1	1	1	1	1	Complement carry
DAA	0	0	1	0	0	1	1	1	Decimal adjust A
CONTROL									
EI	1	1	1	1	1	0	1	1	Enable Interrupts
DI	1	1	1	1	0	0	1	1	Disable Interrupt
NOP	0	0	0	0	0	0	0	0	No-operation
HLT	0	1	1	1	0	1	1	0	Halt
NEW 8085A INSTRUCTIONS									
RIM	0	0	1	0	0	0	0	0	Read Interrupt Mask
SIM	0	0	1	1	0	0	0	0	Set Interrupt Mask

NOTES:

1. DDS or SSS: B 000, C 001, D 010, E011, H 100, L 101, Memory 110, A 111.
2. Two possible cycle times (6/12) indicate instruction cycles dependent on condition flags.

*All mnemonics copyrighted ©Intel Corporation 1976.

8085
INSTRUCTION SET*

A computer, no matter how sophisticated, can only do what it is "told" to do. One "tells" the computer what to do via a series of coded instructions referred to as a **Program**. The realm of the programmer is referred to as **Software**, in contrast to the **Hardware** that comprises the actual computer equipment. A computer's software refers to all of the programs that have been written for that computer.

When a computer is designed, the engineers provide the Central Processing Unit (CPU) with the ability to perform a particular set of operations. The CPU is designed such that a specific operation is performed when the CPU control logic decodes a particular instruction. Consequently, the operations that can be performed by a CPU define the computer's **Instruction Set**.

Each computer instruction allows the programmer to initiate the performance of a specific operation. All computers implement certain arithmetic operations in their instruction set, such as an instruction to add the contents of two registers. Often logical operations (e.g., OR the contents of two registers) and register operate instructions (e.g., increment a register) are included in the instruction set. A computer's instruction set will also have instructions that move data between registers, between a register and memory, and between a register and an I/O device. Most instruction sets also provide **Conditional Instructions**. A conditional instruction specifies an operation to be performed only if certain conditions have been met: for example, jump to a particular instruction if the result of the last operation was zero. Conditional instructions provide a program with a decision-making capability.

By logically organizing a sequence of instructions into a coherent program, the programmer can "tell" the computer to perform a very specific and useful function.

The computer, however, can only execute programs whose instructions are in a binary coded form (i.e., a series of 1's and 0's), that is called **Machine Code**. Because it would be extremely cumbersome to program in machine code, programming languages have been developed. There are programs available which convert the programming language instructions into machine code that can be interpreted by the processor.

One type of programming language is **Assembly Language**. A unique assembly language mnemonic is assigned to each of the computer's instructions. The programmer can write a program (called the **Source Program**) using these mnemonics and certain operands; the source program is then converted into machine instructions (called the **Object Code**). Each assembly language instruction is converted into one machine code instruction (1 or more bytes) by an **Assembler** program. Assembly languages are usually machine dependent (i.e., they are usually able to run on only one type of computer).

THE 8085 INSTRUCTION SET

The 8085 instruction set includes five different types of instructions:

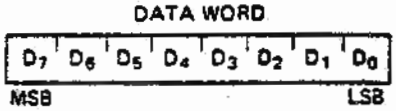
- **Data Transfer Group**—move data between registers or between memory and registers
- **Arithmetic Group** — add, subtract, increment or decrement data in registers or in memory
- **Logical Group** — AND, OR, EXCLUSIVE-OR, compare, rotate or complement data in registers or in memory
- **Branch Group** — conditional and unconditional jump instructions, subroutine call instructions and return instructions
- **Stack, I/O and Machine Control Group** — includes I/O instructions, as well as instructions for maintaining the stack and internal control flags.

Instruction and Data Formats:

Memory for the 8085 is organized into 8-bit quantities, called Bytes. Each byte has a unique 16-bit binary address corresponding to its sequential position in memory.

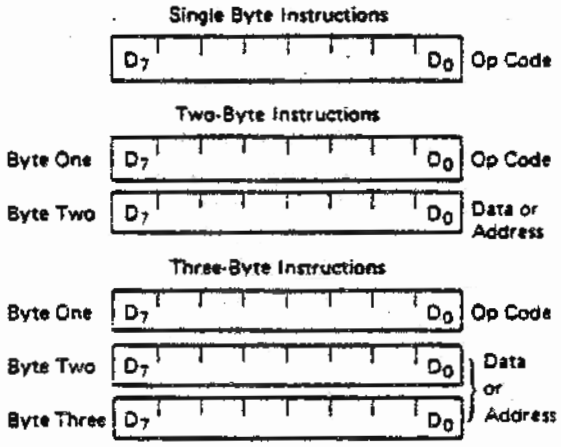
The 8085 can directly address up to 65,536 bytes of memory, which may consist of both read-only memory (ROM) elements and random-access memory (RAM) elements (read/write memory).

Data in the 8085 is stored in the form of 8-bit binary integers:



When a register or data word contains a binary number, it is necessary to establish the order in which the bits of the number are written. In the Intel 8085, BIT 0 is referred to as the **Least Significant Bit (LSB)**, and BIT 7 (of an 8 bit number) is referred to as the **Most Significant Bit (MSB)**.

The 8085 program instructions may be one, two or three bytes in length. Multiple byte instructions must be stored in successive memory locations; the address of the first byte is always used as the address of the instructions. The exact instruction format will depend on the particular operation to be executed.



Addressing Modes:

Often the data that is to be operated on is stored in memory. When multi-byte numeric data is used, the data, like instructions, is stored in successive memory locations, with the least significant byte first, followed by increasingly significant bytes. The 8085 has four different modes for addressing data stored in memory or in registers:

- Direct – Bytes 2 and 3 of the instruction contain the exact memory address of the data item (the low-order bits of the address are in byte 2, the high-order bits in byte 3).
- Register – The instruction specifies the register or register-pair in which the data is located.
- Register Indirect – The instruction specifies a register-pair which contains the memory address where the data is located (the high-order bits of the address are in the first register of the pair, the low-order bits in the second).
- Immediate – The instruction contains the data itself. This is either an 8-bit quantity or a 16-bit quantity (least significant byte first, most significant byte second).

Unless directed by an interrupt or branch instruction, the execution of instructions proceeds through consecutively increasing memory locations. A branch instruction can specify the address of the next instruction to be executed in one of two ways:

- Direct – The branch instruction contains the address of the next instruction to be executed. (Except for the 'RST' instruction, byte 2 contains the low-order address and byte 3 the high-order address.)
- Register indirect – The branch instruction indicates a register-pair which contains the address of the next instruction to be executed. (The high-order bits of the address are in the first register of the pair, the low-order bits in the second.)

The RST instruction is a special one-byte call instruction (usually used during interrupt sequences). RST includes a three-bit field; program control is transferred to the instruction whose address is eight times the contents of this three-bit field.

Condition Flags:

There are five condition flags associated with the execution of instructions on the 8085. They are Zero, Sign, Parity, Carry, and Auxiliary Carry, and are each represented by a 1-bit register in the CPU. A flag is "set" by forcing the bit to 1; "reset" by forcing the bit to 0.

Unless indicated otherwise, when an instruction affects a flag, it affects it in the following manner:

Zero: If the result of an instruction has the value 0, this flag is set; otherwise it is reset.

Sign: If the most significant bit of the result of the operation has the value 1, this flag is set; otherwise it is reset.

Parity: If the modulo 2 sum of the bits of the result of the operation is 0, (i.e., if the result has even parity), this flag is set; otherwise it is reset (i.e., if the result has odd parity).

Carry: If the instruction resulted in a carry (from addition), or a borrow (from subtraction or a comparison) out of the high-order bit, this flag is set; otherwise it is reset.

Auxiliary Carry: If the instruction caused a carry out of bit 3 and into bit 4 of the resulting value, the auxiliary carry is set; otherwise it is reset. This flag is affected by single precision additions, subtractions, increments, decrements, comparisons, and logical operations, but is principally used with additions and increments preceding a DAA (Decimal Adjust Accumulator) instruction.

Symbols and Abbreviations:

The following symbols and abbreviations are used in the subsequent description of the 8085 instructions:

SYMBOLS	MEANING
accumulator	Register A
addr	16-bit address quantity
data	8-bit data quantity
data 16	16-bit data quantity
byte 2	The second byte of the instruction
byte 3	The third byte of the instruction
port	8-bit address of an I/O device
r,r1,r2	One of the registers A,B,C,D,E,H,L
DDD,SSS	The bit pattern designating one of the registers A,B,C,D,E,H,L (DDD=destination, SSS=source):

DDD or SSS	REGISTER NAME
111	A
000	B
001	C
010	D
011	E
100	H
101	L

rp	One of the register pairs: B represents the B,C pair with B as the high-order register and C as the low-order register; D represents the D,E pair with D as the high-order register and E as the low-order register; H represents the H,L pair with H as the high-order register and L as the low-order register; SP represents the 16-bit stack pointer register.
RP	The bit pattern designating one of the register pairs B,D,H,SP:

RP	REGISTER PAIR
00	B-C
01	D-E
10	H-L
11	SP

rh	The first (high-order) register of a designated register pair.
rl	The second (low-order) register of a designated register pair.
PC	16-bit program counter register (PCH and PCL are used to refer to the high-order and low-order 8 bits respectively).
SP	16-bit stack pointer register (SPH and SPL are used to refer to the high-order and low-order 8 bits respectively).
r_m	Bit m of the register r (bits are number 7 through 0 from left to right).
Z,S,P,CY,AC	The condition flags: Zero, Sign, Parity, Carry, and Auxiliary Carry, respectively.
( )	The contents of the memory location or registers enclosed in the parentheses.
←	"Is transferred to"
∧	Logical AND
∀	Exclusive OR
∨	Inclusive OR
+	Addition
−	Two's complement subtraction
*	Multiplication
← →	"Is exchanged with"
$\overline{\ }$	The one's complement (e.g., $(\overline{A})$)
n	The restart number 0 through 7
NNN	The binary representation 000 through 111 for restart number 0 through 7 respectively.

Description Format:

The following pages provide a detailed description of the instruction set of the 8085. Each instruction is described in the following manner:

1. The MCS 85™ macro assembler format, consisting of the instruction mnemonic and operand fields, is printed in BOLDFACE on the left side of the first line.
2. The name of the instruction is enclosed in parenthesis on the right side of the first line.
3. The next line(s) contain a symbolic description of the operation of the instruction.
4. This is followed by a narative description of the operation of the instruction.
5. The following line(s) contain the binary fields and patterns that comprise the machine instruction.

6. The last four lines contain incidental information about the execution of the instruction. The number of machine cycles and states required to execute the instruction are listed first. If the instruction has two possible execution times, as in a Conditional Jump, both times will be listed, separated by a slash. Next, any significant data addressing modes (see Page 4-2) are listed. The last line lists any of the five Flags that are affected by the execution of the instruction.

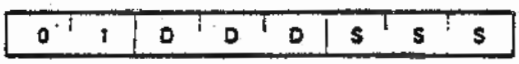
Data Transfer Group:

This group of instructions transfers data to and from registers and memory. Condition flags are not affected by any instruction in this group.

MOV r1, r2 (Move Register)

(r1) ← (r2)

The content of register r2 is moved to register r1.

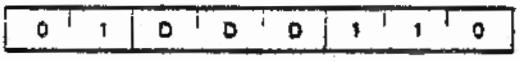


Cycles: 1
States: 4
Addressing: register
Flags: none

MOV r, M (Move from memory)

(r) ← ((H) (L))

The content of the memory location, whose address is in registers H and L, is moved to register r.

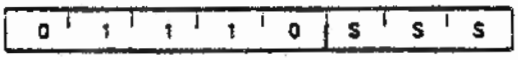


Cycles: 2
States: 7
Addressing: reg. indirect
Flags: none

MOV M, r (Move to memory)

((H) (L)) ← (r)

The content of register r is moved to the memory location whose address is in registers H and L.

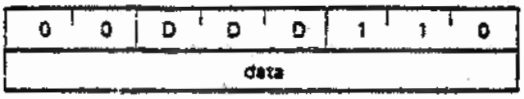


Cycles: 2
States: 7
Addressing: reg. indirect
Flags: none

MVI r, data (Move Immediate)

(r) ← (byte 2)

The content of byte 2 of the instruction is moved to register r.

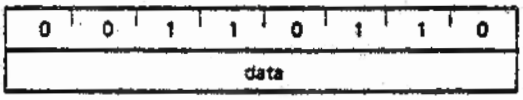


Cycles: 2
States: 7
Addressing: immediate
Flags: none

MVI M, data (Move to memory immediate)

((H) (L)) ← (byte 2)

The content of byte 2 of the instruction is moved to the memory location whose address is in registers H and L.

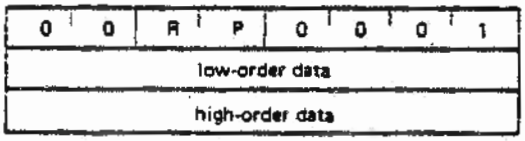


Cycles: 3
States: 10
Addressing: immed./reg. indirect
Flags: none

LXI rp, data 16 (Load register pair immediate)

(rh) ← (byte 3),
(rl) ← (byte 2)

Byte 3 of the instruction is moved into the high-order register (rh) of the register pair rp. Byte 2 of the instruction is moved into the low-order register (rl) of the register pair rp.

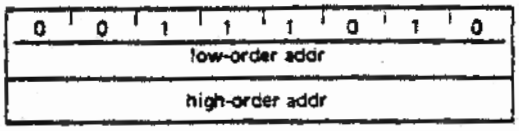


Cycles: 3
States: 10
Addressing: immediate
Flags: none

LDA addr (Load Accumulator direct)

(A) ← ((byte 3)(byte 2))

The content of the memory location, whose address is specified in byte 2 and byte 3 of the instruction, is moved to register A.

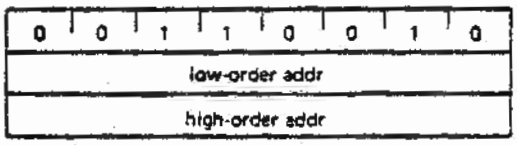


Cycles: 4
States: 13
Addressing: direct
Flags: none

STA addr (Store Accumulator direct)

((byte 3)(byte 2)) ← (A)

The content of the accumulator is moved to the memory location whose address is specified in byte 2 and byte 3 of the instruction.



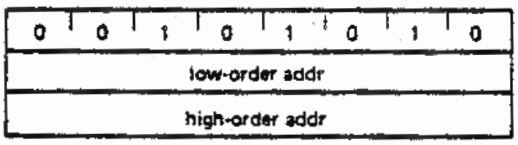
Cycles: 4
States: 13
Addressing: direct
Flags: none

LHLD addr (Load H and L direct)

(L) ← ((byte 3)(byte 2))

(H) ← ((byte 3)(byte 2) + 1)

The content of the memory location, whose address is specified in byte 2 and byte 3 of the instruction, is moved to register L. The content of the memory location at the succeeding address is moved to register H.



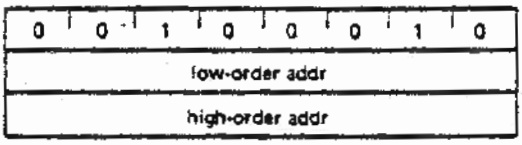
Cycles: 5
States: 16
Addressing: direct
Flags: none

SHLD addr (Store H and L direct)

((byte 3)(byte 2)) ← (L)

((byte 3)(byte 2) + 1) ← (H)

The content of register L is moved to the memory location whose address is specified in byte 2 and byte 3. The content of register H is moved to the succeeding memory location.



Cycles: 5
States: 16
Addressing: direct
Flags: none

LDAX rp (Load accumulator indirect)

(A) ← ((rp))

The content of the memory location, whose address is in the register pair rp, is moved to register A. Note: only register pairs rp=B (registers B and C) or rp=D (registers D and E) may be specified.



Cycles: 2
States: 7
Addressing: reg. indirect
Flags: none

STAX rp (Store accumulator indirect)

((rp)) ← (A)

The content of register A is moved to the memory location whose address is in the register pair rp. Note: only register pairs rp=B (registers B and C) or rp=D (registers D and E) may be specified.



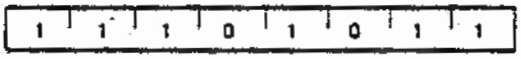
Cycles: 2
States: 7
Addressing: reg. indirect
Flags: none

XCHG (Exchange H and L with D and E)

(H) ⟷ (D)

(L) ⟷ (E)

The contents of registers H and L are exchanged with the contents of registers D and E.



Cycles: 1
States: 4
Addressing: register
Flags: none

Arithmetic Group:

This group of instructions performs arithmetic operations on data in registers and memory.

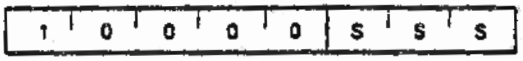
Unless indicated otherwise, all instructions in this group affect the Zero, Sign, Parity, Carry, and Auxiliary Carry flags according to the standard rules.

All subtraction operations are performed via two's complement arithmetic and set the carry flag to one to indicate a borrow and clear it to indicate no borrow.

ADD r (Add Register)

(A) ← (A) + (r)

The content of register r is added to the content of the accumulator. The result is placed in the accumulator.

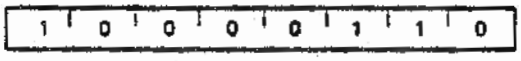


Cycles: 1
States: 4
Addressing: register
Flags: Z,S,P,CY,AC

ADD M (Add memory)

(A) ← (A) + ((H) (L))

The content of the memory location whose address is contained in the H and L registers is added to the content of the accumulator. The result is placed in the accumulator.

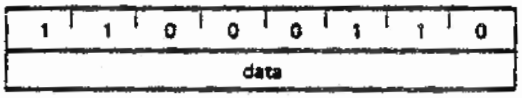


Cycles: 2
States: 7
Addressing: reg. indirect
Flags: Z,S,P,CY,AC

ADI data (Add immediate)

(A) ← (A) + (byte 2)

The content of the second byte of the instruction is added to the content of the accumulator. The result is placed in the accumulator.

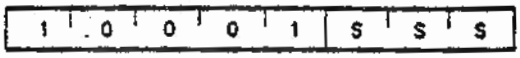


Cycles: 2
States: 7
Addressing: immediate
Flags: Z,S,P,CY,AC

ADC r (Add Register with carry)

(A) ← (A) + (r) + (CY)

The content of register r and the content of the carry bit are added to the content of the accumulator. The result is placed in the accumulator.

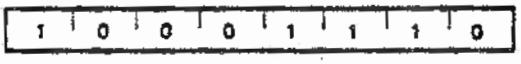


Cycles: 1
States: 4
Addressing: register
Flags: Z,S,P,CY,AC

ADC M (Add memory with carry)

(A) ← (A) + ((H) (L)) + (CY)

The content of the memory location whose address is contained in the H and L registers and the content of the CY flag are added to the accumulator. The result is placed in the accumulator.

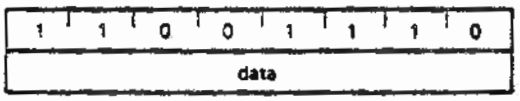


Cycles: 2
States: 7
Addressing: reg. indirect
Flags: Z,S,P,CY,AC

ACI data (Add immediate with carry)

(A) ← (A) + (byte 2) + (CY)

The content of the second byte of the instruction and the content of the CY flag are added to the contents of the accumulator. The result is placed in the accumulator.

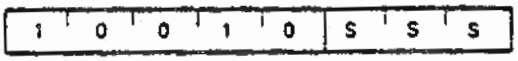


Cycles: 2
States: 7
Addressing: immediate
Flags: Z,S,P,CY,AC

SUB r (Subtract Register)

(A) ← (A) − (r)

The content of register r is subtracted from the content of the accumulator. The result is placed in the accumulator.

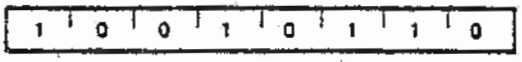


Cycles: 1
States: 4
Addressing: register
Flags: Z,S,P,CY,AC

SUB M (Subtract memory)

(A) ← (A) − ((H) (L))

The content of the memory location whose address is contained in the H and L registers is subtracted from the content of the accumulator. The result is placed in the accumulator.

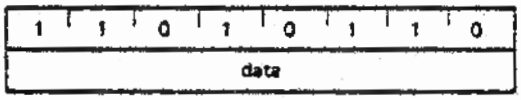


Cycles: 2
States: 7
Addressing: reg. indirect
Flags: Z,S,P,CY,AC

SUI data (Subtract immediate)

(A) ← (A) − (byte 2)

The content of the second byte of the instruction is subtracted from the content of the accumulator. The result is placed in the accumulator.

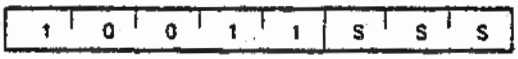


Cycles: 2
States: 7
Addressing: immediate
Flags: Z,S,P,CY,AC

SBB r (Subtract Register with borrow)

(A) ← (A) − (r) − (CY)

The content of register r and the content of the CY flag are both subtracted from the accumulator. The result is placed in the accumulator.

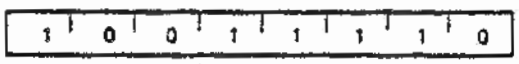


Cycles: 1
States: 4
Addressing: register
Flags: Z,S,P,CY,AC

SBB M (Subtract memory with borrow)

(A) ← (A) − ((H) (L)) − (CY)

The content of the memory location whose address is contained in the H and L registers and the content of the CY flag are both subtracted from the accumulator. The result is placed in the accumulator.

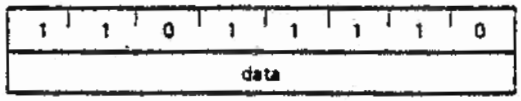


Cycles: 2
States: 7
Addressing: reg. indirect
Flags: Z,S,P,CY,AC

SBI data (Subtract immediate with borrow)

(A) ← (A) − (byte 2) − (CY)

The contents of the second byte of the instruction and the contents of the CY flag are both subtracted from the accumulator. The result is placed in the accumulator.

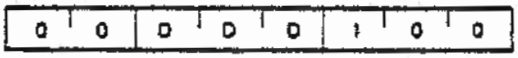


Cycles: 2
States: 7
Addressing: immediate
Flags: Z,S,P,CY,AC

INR r (Increment Register)

(r) ← (r) + 1

The content of register r is incremented by one. Note: All condition flags except CY are affected.

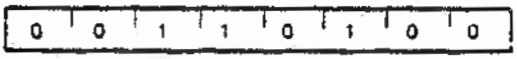


Cycles: 1
States: 4
Addressing: register
Flags: Z,S,P,AC

INR M (Increment memory)

((H) (L)) ← ((H) (L)) + 1

The content of the memory location whose address is contained in the H and L registers is incremented by one. Note: All condition flags except CY are affected.

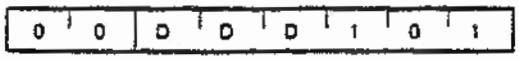


Cycles: 3
States: 10
Addressing: reg. indirect
Flags: Z,S,P,AC

DCR r (Decrement Register)

(r) ← (r) − 1

The content of register r is decremented by one. Note: All condition flags except CY are affected.

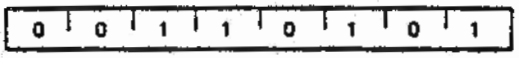


Cycles: 1
States: 4
Addressing: register
Flags: Z,S,P,AC

DCR M (Decrement memory)

((H) (L)) ← ((H) (L)) – 1

The content of the memory location whose address is contained in the H and L registers is decremented by one. Note: All condition flags except CY are affected.



Cycles: 3
States: 10
Addressing: reg. indirect
Flags: Z,S,P,AC

INX rp (Increment register pair)

(rh) (rl) ← (rh) (rl) + 1

The content of the register pair rp is incremented by one. Note: No condition flags are affected.



Cycles: 1
States: 6
Addressing: register
Flags: none

DCX rp (Decrement register pair)

(rh) (rl) ← (rh) (rl) – 1

The content of the register pair rp is decremented by one. Note: No condition flags are affected.



Cycles: 1
States: 6
Addressing: register
Flags: none

DAD rp (Add register pair to H and L)

(H) (L) ← (H) (L) + (rh) (rl)

The content of the register pair rp is added to the content of the register pair H and L. The result is placed in the register pair H and L. Note: Only the CY flag is affected. It is set if there is a carry out of the double precision add; otherwise it is reset.



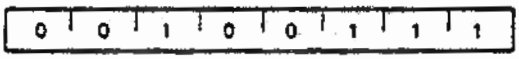
Cycles: 3
States: 10
Addressing: register
Flags: CY

DAA (Decimal Adjust Accumulator)

The eight-bit number in the accumulator is adjusted to form two four-bit Binary-Coded-Decimal digits by the following process:

1. If the value of the least significant 4 bits of the accumulator is greater than 9 or if the AC flag is set, 6 is added to the accumulator.
2. If the value of the most significant 4 bits of the accumulator is now greater than 9, or if the CY flag is set, 6 is added to the most significant 4 bits of the accumulator.

NOTE: All flags are affected.



Cycles: 1
States: 4
Flags: Z,S,P,CY,AC

Logical Group:

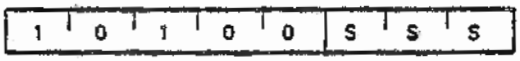
This group of instructions performs logical (Boolean) operations on data in registers and memory and on condition flags.

Unless indicated otherwise, all instructions in this group affect the Zero, Sign, Parity, Auxiliary Carry, and Carry flags according to the standard rules.

ANA r (AND Register)

(A) ← (A) ∧ (r)

The content of register r is logically anded with the content of the accumulator. The result is placed in the accumulator. The CY flag is cleared and AC is set.



Cycles: 1
States: 4
Addressing: register
Flags: Z,S,P,CY,AC

ANA M (AND memory)

(A) ← (A) ∧ ((H) (L))

The contents of the memory location whose address is contained in the H and L registers is logically anded with the content of the accumulator. The result is placed in the accumulator. The CY flag is cleared and AC is set.

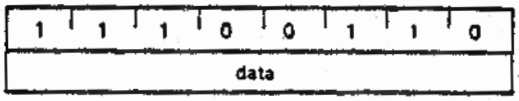


Cycles: 2
States: 7
Addressing: reg. indirect
Flags: Z,S,P,CY,AC

ANI data (AND immediate)

(A) ← (A) ∧ (byte 2)

The content of the second byte of the instruction is logically anded with the contents of the accumulator. The result is placed in the accumulator. The CY flag is cleared and AC is set.

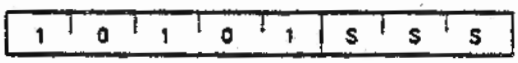


Cycles: 2
States: 7
Addressing: immediate
Flags: Z,S,P,CY,AC

XRA r (Exclusive OR Register)

(A) ← (A) ∀ (r)

The content of register r is exclusive-or'd with the content of the accumulator. The result is placed in the accumulator. The CY and AC flags are cleared.

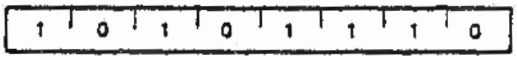


Cycles: 1
States: 4
Addressing: register
Flags: Z,S,P,CY,AC

XRA M (Exclusive OR Memory)

(A) ← (A) ∀ ((H) (L))

The content of the memory location whose address is contained in the H and L registers is exclusive-OR'd with the content of the accumulator. The result is placed in the accumulator. The CY and AC flags are cleared.

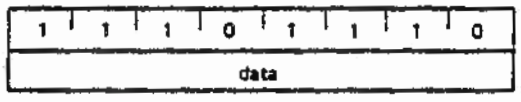


Cycles: 2
States: 7
Addressing: reg. indirect
Flags: Z,S,P,CY,AC

XRI data (Exclusive OR immediate)

(A) ← (A) ∀ (byte 2)

The content of the second byte of the instruction is exclusive-OR'd with the content of the accumulator. The result is placed in the accumulator. The CY and AC flags are cleared.

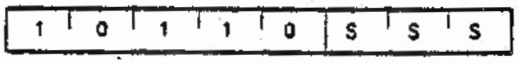


Cycles: 2
States: 7
Addressing: immediate
Flags: Z,S,P,CY,AC

ORA r (OR Register)

(A) ← (A) ∨ (r)

The content of register r is inclusive-OR'd with the content of the accumulator. The result is placed in the accumulator. The CY and AC flags are cleared.

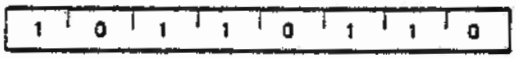


Cycles: 1
States: 4
Addressing: register
Flags: Z,S,P,CY,AC

ORA M (OR memory)

(A) ← (A) ∨ ((H) (L))

The content of the memory location whose address is contained in the H and L registers is inclusive-OR'd with the content of the accumulator. The result is placed in the accumulator. The CY and AC flags are cleared.

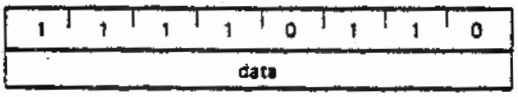


Cycles: 2
States: 7
Addressing: reg. indirect
Flags: Z,S,P,CY,AC

ORI data (OR Immediate)

(A) ← (A) ∨ (byte 2)

The content of the second byte of the instruction is inclusive-OR'd with the content of the accumulator. The result is placed in the accumulator. The CY and AC flags are cleared.

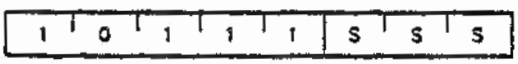


Cycles: 2
States: 7
Addressing: immediate
Flags: Z,S,P,CY,AC

CMP r (Compare Register)

(A) − (r)

The content of register r is subtracted from the accumulator. The accumulator remains unchanged. The condition flags are set as a result of the subtraction. The Z flag is set to 1 if (A) = (r). The CY flag is set to 1 if (A) < (r).



Cycles: 1
States: 4
Addressing: register
Flags: Z,S,P,CY,AC

CMP M (Compare memory)

(A) − ((H) (L))

The content of the memory location whose address is contained in the H and L registers is subtracted from the accumulator. The accumulator remains unchanged. The condition flags are set as a result of the subtraction. The Z flag is set to 1 if (A) = ((H) (L)). The CY flag is set to 1 if (A) < ((H) (L)).



Cycles: 2
States: 7
Addressing: reg. indirect
Flags: Z,S,P,CY,AC

CPI data (Compare immediate)

(A) − (byte 2)

The content of the second byte of the instruction is subtracted from the accumulator. The condition flags are set by the result of the subtraction. The Z flag is set to 1 if (A) = (byte 2). The CY flag is set to 1 if (A) < (byte 2).



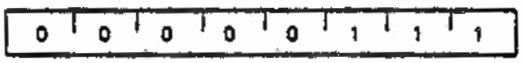
Cycles: 2
States: 7
Addressing: immediate
Flags: Z,S,P,CY,AC

RLC (Rotate left)

$(A_{n+1}) \leftarrow (A_n)$; $(A_0) \leftarrow (A_7)$

$(CY) \leftarrow (A_7)$

The content of the accumulator is rotated left one position. The low order bit and the CY flag are both set to the value shifted out of the high order bit position. Only the CY flag is affected.



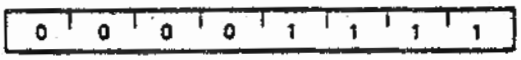
Cycles: 1
States: 4
Flags: CY

RRC (Rotate right)

$(A_n) \leftarrow (A_{n-1})$; $(A_7) \leftarrow (A_0)$

$(CY) \leftarrow (A_0)$

The content of the accumulator is rotated right one position. The high order bit and the CY flag are both set to the value shifted out of the low order bit position. Only the CY flag is affected.



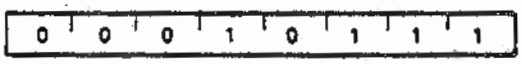
Cycles: 1
States: 4
Flags: CY

RAL (Rotate left through carry)

$(A_{n+1}) \leftarrow (A_n)$; $(CY) \leftarrow (A_7)$

$(A_0) \leftarrow (CY)$

The content of the accumulator is rotated left one position through the CY flag. The low order bit is set equal to the CY flag and the CY flag is set to the value shifted out of the high order bit. Only the CY flag is affected.



Cycles: 1
States: 4
Flags: CY

RAR (Rotate right through carry)

$(A_n) \leftarrow (A_{n+1})$; $(CY) \leftarrow (A_0)$

$(A_7) \leftarrow (CY)$

The content of the accumulator is rotated right one position through the CY flag. The high order bit is set to the CY flag and the CY flag is set to the value shifted out of the low order bit. Only the CY flag is affected.

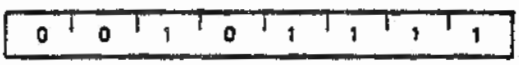


Cycles: 1
States: 4
Flags: CY

CMA (Complement accumulator)

$(A) \leftarrow (\overline{A})$

The contents of the accumulator are complemented (zero bits become 1, one bits become 0). No flags are affected.



Cycles: 1
States: 4
Flags: none

CMC (Complement carry)

(CY) ← $\overline{(CY)}$

The CY flag is complemented. No other flags are affected.



Cycles: 1
States: 4
Flags: CY

STC (Set carry)

(CY) ← 1

The CY flag is set to 1. No other flags are affected.



Cycles: 1
States: 4
Flags: CY

Branch Group:

This group of instructions alter normal sequential program flow.

Condition flags are not affected by any instruction in this group.

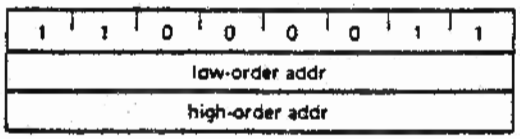
The two types of branch instructions are unconditional and conditional. Unconditional transfers simply perform the specified operation on register PC (the program counter). Conditional transfers examine the status of one of the four processor flags to determine if the specified branch is to be executed. The conditions that may be specified are as follows:

CONDITION	CCC
NZ – not zero (Z = 0)	000
Z – zero (Z = 1)	001
NC – no carry (CY = 0)	010
C – carry (CY = 1)	011
PO – parity odd (P = 0)	100
PE – parity even (P = 1)	101
P – plus (S = 0)	110
M – minus (S = 1)	111

JMP addr (Jump)

(PC) ← (byte 3) (byte 2)

Control is transferred to the instruction whose address is specified in byte 3 and byte 2 of the current instruction.



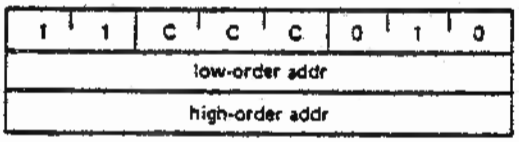
Cycles: 3
States: 10
Addressing: immediate
Flags: none

Jcondition addr (Conditional jump)

If (CCC),

(PC) ← (byte 3) (byte 2)

If the specified condition is true, control is transferred to the instruction whose address is specified in byte 3 and byte 2 of the current instruction; otherwise, control continues sequentially.



Cycles: 2/3
States: 7/10
Addressing: immediate
Flags: none

CALL addr (Call)

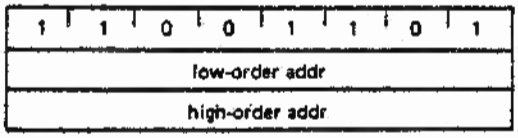
((SP) – 1) ← (PCH)

((SP) – 2) ← (PCL)

(SP) ← (SP) – 2

(PC) ← (byte 3) (byte 2)

The high-order eight bits of the next instruction address are moved to the memory location whose address is one less than the content of register SP. The low-order eight bits of the next instruction address are moved to the memory location whose address is two less than the content of register SP. The content of register SP is decremented by 2. Control is transferred to the instruction whose address is specified in byte 3 and byte 2 of the current instruction.



Cycles: 5
States: 18
Addressing: immediate/reg. indirect
Flags: none

Ccondition addr (Condition call)

If (CCC),
(SP) − 1) ← (PCH)
((SP) − 2) ← (PCL)
(SP) ← (SP) − 2
(PC) ← (byte 3) (byte 2)

If the specified condition is true, the actions specified in the CALL instruction (see above) are performed; otherwise, control continues sequentially.

1	1	C	C	C	1	0	0
low-order addr							
high-order addr							

Cycles: 2/5
States: 9/18
Addressing: immediate/reg. indirect
Flags: none

RET (Return)

(PCL) ← ((SP));
(PCH) ← ((SP) + 1);
(SP) ← (SP) + 2;

The content of the memory location whose address is specified in register SP is moved to the low-order eight bits of register PC. The content of the memory location whose address is one more than the content of register SP is moved to the high-order eight bits of register PC. The content of register SP is incremented by 2.

1	1	0	0	1	0	0	1

Cycles: 3
States: 10
Addressing: reg. indirect
Flags: none

Rcondition (Conditional return)

If (CCC),
(PCL) ← ((SP))
(PCH) ← ((SP) + 1)
(SP) ← (SP) + 2

If the specified condition is true, the actions specified in the RET instruction (see above) are performed; otherwise, control continues sequentially.

1	1	C	C	C	0	0	0

Cycles: 1/3
States: 6/12
Addressing: reg. indirect
Flags: none

RST n (Restart)

((SP) − 1) ← (PCH)
((SP) − 2) ← (PCL)
(SP) ← (SP) − 2
(PC) ← 8 * (NNN)

The high-order eight bits of the next instruction address are moved to the memory location whose address is one less than the content of register SP. The low-order eight bits of the next instruction address are moved to the memory location whose address is two less than the content of register SP. The content of register SP is decremented by two. Control is transferred to the instruction whose address is eight times the content of NNN.

1	1	N	N	N	1	1	1

Cycles: 3
States: 12
Addressing: reg. indirect
Flags: none

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	N	N	N	0	0	0

Program Counter After Restart

PCHL (Jump H and L indirect – move H and L to PC)

(PCH) ← (H)
(PCL) ← (L)

The content of register H is moved to the high-order eight bits of register PC. The content of register L is moved to the low-order eight bits of register PC.

1	1	1	0	1	0	0	1

Cycles: 1
States: 6
Addressing: register
Flags: none

Stack, I/O, and Machine Control Group:

This group of instructions performs I/O, manipulates the Stack, and alters internal control flags.

Unless otherwise specified, condition flags are not affected by any instructions in this group.

PUSH rp (Push)

((SP) − 1) ← (rh)
((SP) − 2) ← (rl)
(SP) ← (SP) − 2

The content of the high-order register of register pair rp is moved to the memory location whose address is one less than the content of register SP. The content of the low-order register of register pair rp is moved to the memory location whose address is two less than the content of register SP. The content of register SP is decremented by 2. Note: Register pair rp = SP may not be specified.

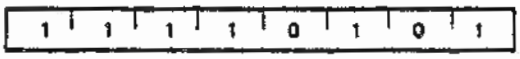


Cycles: 3
States: 12
Addressing: reg. indirect
Flags: none

PUSH PSW (Push processor status word)

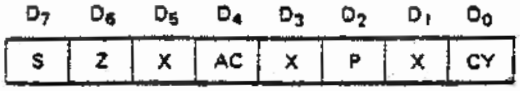
((SP) − 1) ← (A)
((SP) − 2)$_0$ ← (CY), ((SP) − 2)$_1$ ← 1
((SP) − 2)$_2$ ← (P), ((SP) − 2)$_3$ ← 0
((SP) − 2)$_4$ ← (AC), ((SP) − 2)$_5$ ← 0
((SP) − 2)$_6$ ← (Z), ((SP) − 2)$_7$ ← (S)
(SP) ← (SP) − 2

The content of register A is moved to the memory location whose address is one less than register SP. The contents of the condition flags are assembled into a processor status word and the word is moved to the memory location whose address is two less than the content of register SP. The content of register SP is decremented by two.



Cycles: 3
States: 12
Addressing: reg. indirect
Flags: none

FLAG WORD



X: undefined

POP rp (Pop)

(rl) ← ((SP))
(rh) ← ((SP) + 1)
(SP) ← (SP) + 2

The content of the memory location, whose address is specified by the content of register SP, is moved to the low-order register of register pair rp. The content of the memory location, whose address is one more than the content of register SP, is moved to the high-order register of register pair rp. The content of register SP is incremented by 2. Note: Register pair rp = SP may not be specified.

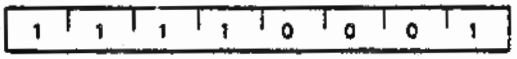


Cycles: 3
States: 10
Addressing: reg. indirect
Flags: none

POP PSW (Pop processor status word)

(CY) ← ((SP))$_0$
(P) ← ((SP))$_2$
(AC) ← ((SP))$_4$
(Z) ← ((SP))$_6$
(S) ← ((SP))$_7$
(A) ← ((SP) + 1)
(SP) ← (SP) + 2

The content of the memory location whose address is specified by the content of register SP is used to restore the condition flags. The content of the memory location whose address is one more than the content of register SP is moved to register A. The content of register SP is incremented by 2.

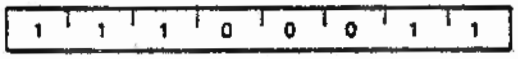


Cycles: 3
States: 10
Addressing: reg. indirect
Flags: Z,S,P,CY,AC

XTHL (Exchange stack top with H and L)

(L) ⟷ ((SP))
(H) ⟷ ((SP) + 1)

The content of the L register is exchanged with the content of the memory location whose address is specified by the content of register SP. The content of the H register is exchanged with the content of the memory location whose address is one more than the content of register SP.

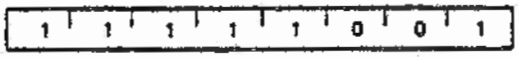


Cycles: 5
States: 16
Addressing: reg. indirect
Flags: none

SPHL (Move HL to SP)

(SP) ⟷ (H) (L)

The contents of registers H and L (16 bits) are moved to register SP.

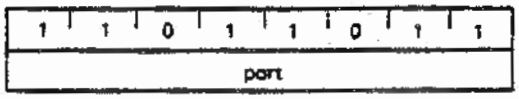


Cycles: 1
States: 6
Addressing: register
Flags: none

IN port (Input)

(A) ⟷ (data)

The data placed on the eight bit bi-directional data bus by the specified port is moved to register A.

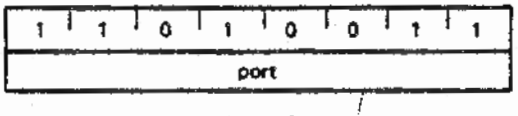


Cycles: 3
States: 10
Addressing: direct
Flags: none

OUT port (Output)

(data) ⟷ (A)

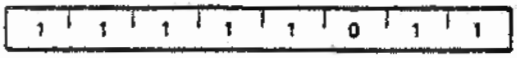
The content of register A is placed on the eight bit bi-directional data bus for transmission to the specified port.



Cycles: 3
States: 10
Addressing: direct
Flags: none

EI (Enable interrupts)

The interrupt system is enabled following the execution of the next instruction.



Cycles: 1
States: 4
Flags: none

DI (Disable interrupts)

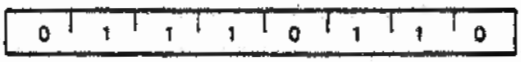
The interrupt system is disabled immediately following the execution of the DI instruction.



Cycles: 1
States: 4
Flags: none

HLT (Halt)

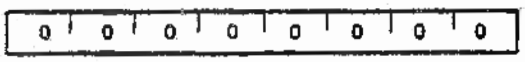
The processor is stopped. The registers and flags are unaffected.



Cycles: 1
States: 5
Flags: none

NOP (No op)

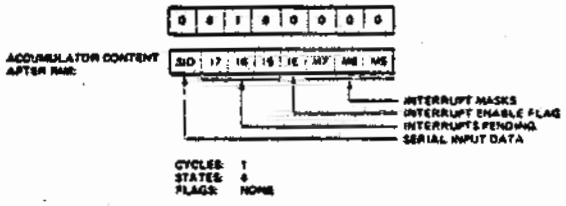
No operation is performed. The registers and flags are unaffected.



Cycles: 1
States: 4
Flags: none

RIM (Read Interrupt Mask)

The accumulator is loaded with the restart interrupt masks, any pending interrupts, and the contents of the serial input data line (SID).



SIM (Set Interrupt Masks)

The contents of the accumulator will be used in programming the restart interrupt masks. Bits 0-2 will set/reset the mask bit for RST 5.5, 6.5, 7.5 of the interrupt mask register, if bit 3 is 1 ("set"). Bit 3 is a "Mask Set Enable" control.

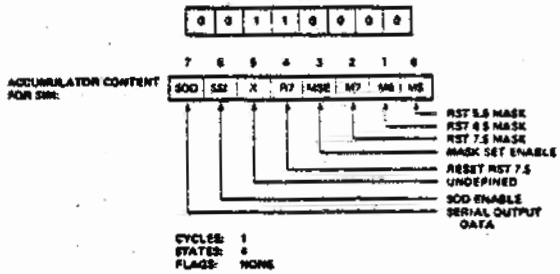
Setting the mask (i.e. masked bit = 1) disables the corresponding interrupt.

	Set	Reset
RST 5.5 MASK	if bit 0 = 1	if bit 0 = 0
RST 6.5 MASK	bit 1 = 1	bit 1 = 0
RST 7.5 MASK	bit 2 = 1	bit 2 = 0

RST 7.5, whether masked or not, will be reset if bit 4 = 1.

$\overline{\text{RESET IN}}$ input (pin 36) will set all RST MASKs, and reset/disable all interrupts.

SIM can, also, load the SOD output latch. Accumulator bit 7 is loaded into the SOD latch if bit 6 is set. The latch is unaffected if bit 6 is a zero. $\overline{\text{RESET IN}}$ input sets the SOD latch to zero.

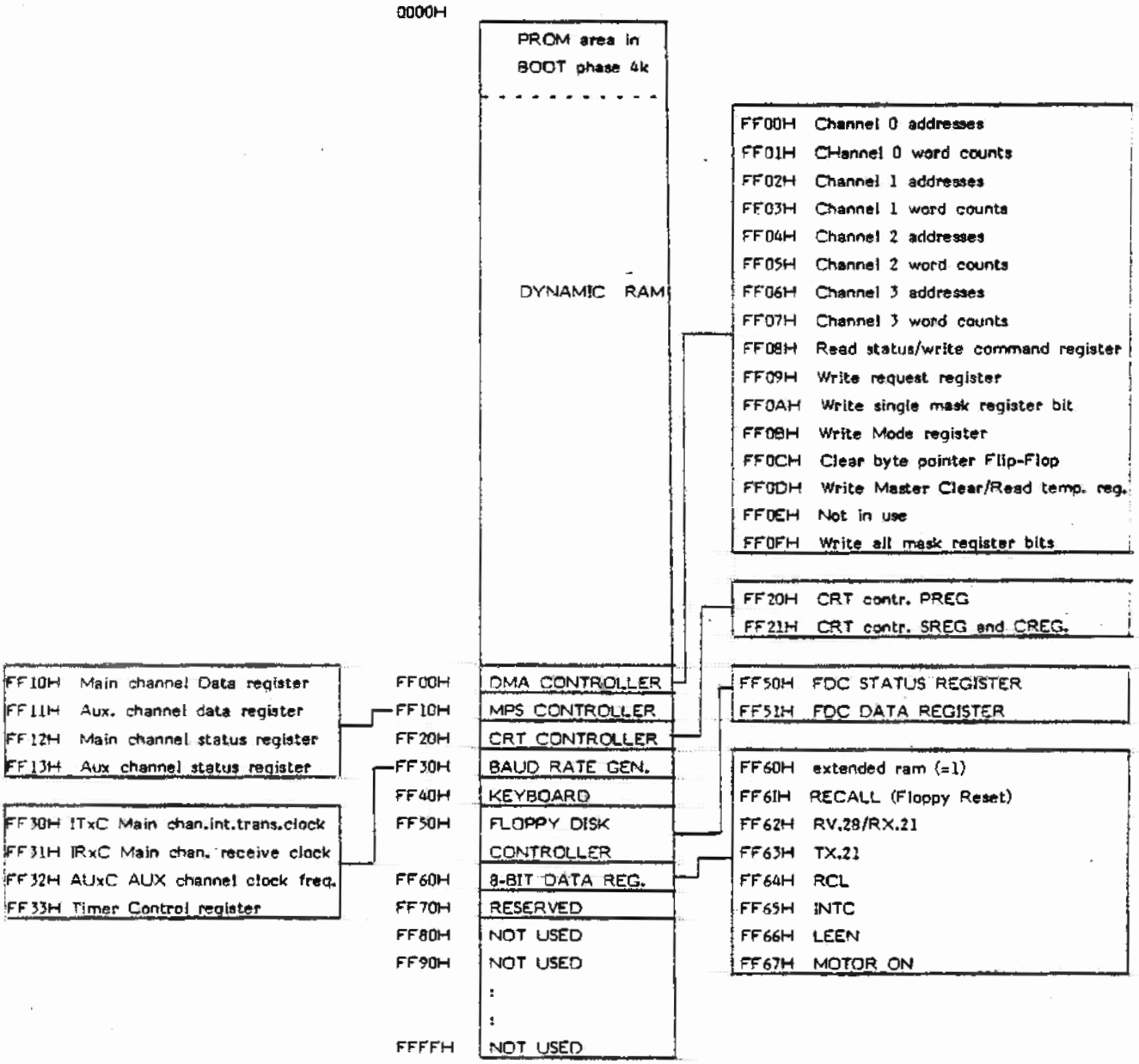


ADRESSING LOGIC

Every peripheral circuit of microprocessor has its own individual address in the address space of the CPU. Physical devices are adressed like memory. Address space from FF00H upwards is reserved to physical devices, with increment of 10H. Lowest 4 kbytes, 0000H-0FFFH is reserved for use either as ROM or RAM. After power reset ROM is selected, and processor starts program execution from memory location 0000H. This ROM-located program contains self-test diagnostic program, and Floppy disk bootstrap loader. After that operating system is loaded, PROM memory is switched off.

Dynamic RAM must be refreshed with intervals less than 2 ms, not to loose its contents. This can be done with READ or WRITE operation or with RAS-only or HIDDEN refresh cycle. READ operation is used in MicroMikko. This is done at the same time with refreshing display with the help of DMA.

Characters to be displayed are located in one queue in the memory, and this memory area is read with 20 ms intervals. This queue is 2000 characters long, it is 25 lines, 80 characters each. DRAM refresh must be done to the 128 lowest adresses (A0-A6) with intervals less than 2 ms. When refreshing display, two lines (=160 char's) covering this 128 bytes, are read during every 1,6 ms, being short enough for refreshing RAM.



MicroMikko Address space

8-BIT DATA REGISTER

The programmable 8-bit dataregister enables changes in the operation of peripheral devices during run time. Data bit 0 is the only one in use. After RESET the logic state of every bit is zero.

Address	data	Name	Operation
FF60	0	EXTENDED	EPROM in use from loc 0000
	1	RAM	DRAM in use from loc 0000
FF61	0	RECALL	Floppydisk controller in use
	1		Floppydisk controller in RESET state
FF62	0	RV.28/RX.21	Receive via V.28/TTL Drivers
	1		Receive via X.21/TTL Drivers
FF63	0	TX.21	TRANSMIT also via X.21 Drivers
	1		TRANSMIT only via V.28 Drivers and set DTR in main channel
FF64	0	RCL	Current loop receive ON
	1		Current loop receive OFF
FF65	0	INTC	Main channel is using internal clock
	1		Main channel is using external clock
FF66	0	LEEN	ASCII-character set in use in display
	1		Semigrafic char. set permitted in display
FF67	0	MOTOR ON	FDD Motor halted
	1		FDD Motor running

INTERRUPTS

INTERRUPT	FROM	Branch address(HEX)
INT	MPSC tells any change in the status, receiving or transmitting information	note 1
RST 5.5	FDC tells CPU being ready after former operation and that it is going to "RESULT PHASE" state.	2C
RST 6.5	KEYBOARD, a new code is ready	34
RST 7.5	DMA-End of process	3C
TRAP	Not in use	24

note 1: MPSC must be programmed to perform this operation

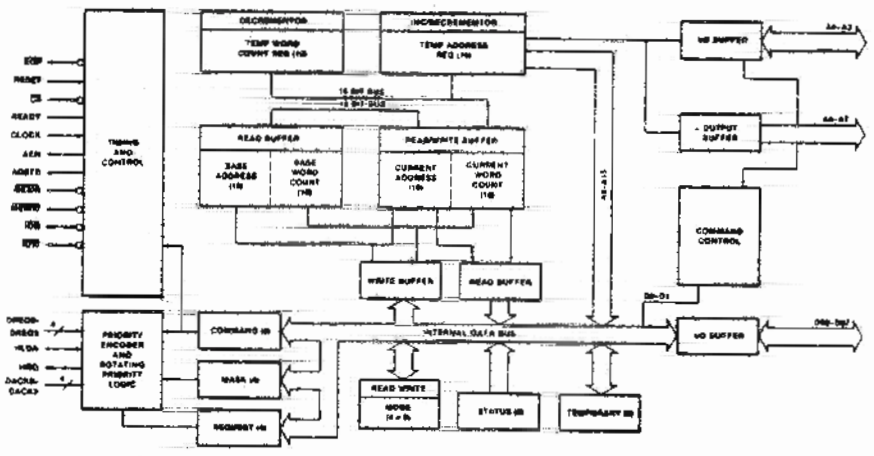
2.2 DMA USAGE AND PROGRAMMING

DMA has a central role in the MicroMikkos operation. It is used to refresh display and dynamic RAM and to transfer data from/to mass storage media and serial communications ports. The DMA-controller is realized with Intels 8237 or similar. The 8237 is a programmable, Direct Memory Access (DMA) device which provides a complete four-channel DMA-controller.

DMA channels are used for

channel 0	crt controller
channel 1	MPSC, Main channel, transmitted data
channel 2	MPSC, Main channel, received data
channel 3	FD controller

This means that DMA controllers Device Request (DREQ) and DMA Acknowledge (DACK) signals are wired to these I/O controllers.



FUNCTIONAL DESCRIPTION

The 8237 block diagram includes the major logic blocks and all of the internal registers. The data interconnection paths are also shown. Not shown are the various control signals between the blocks. The 8237 contains 344 bits of internal memory in the form of registers. Figure lists these registers by name and shows below the size and numbers of each. A detailed description of the registers and their functions can be found under Register Description.

Name	Size	Number
Base Address Register	16 bits	4
Base Word Count Register	16 bits	4
Current Address Register	16 bits	4
Current Word Count Register	16 bits	4
Temporary Address Register	16 bits	1
Temporary Word Count Register	16 bits	1
Status Register	8 bits	1
Command Register	8 bits	1
Temporary Register	8 bits	1
Mode Register	6 bits	4
Mask Register	4 bits	1
Request Register	4 bits	1

8237 Internal Registers

The 8237 contains three basic blocks of control logic. The timing Control block generates internal timing and external control signals for the 8237. The Program Command Control block decodes the various commands given to the 8237 by the microprocessor prior to servicing a DMA Request. It also decodes the Mode Control word used to select the type of DMA during the servicing. The Priority Encoder block resolves priority contention between DMA channels requesting service simultaneously.

The Timing Control block derives internal timing from the clock input. In 8237 systems this input will be the CLK from an 8085A.

DMA OPERATION

The 8237 is designed to operate in two major cycles. These are called Idle and Active cycles. Each device cycle is made up of a number of states. The 8237 can assume seven separate states, each composed of one full clock period. State I (SI) is the inactive state. It is entered when the 8237 has no valid DMA requests pending. While in SI, the DMA controller is inactive but may be in the Program Condition, Being programmed by the processor. State O (SO) is the first state of a DMA service. The 8237 has requested a hold but the processor has not yet returned an acknowledge. An acknowledge from the CPU will signal that transfers may begin. S1, S2, S3 and S4 are the working states of the DMA service.

Memory-to-memory transfer require a read from and a write-to-memory to complete each transfer. The states, which resemble the normal working states, use two digit number for indentification. Eight states are required for a single transfer. The first four states (S11, S12, S13, S14) are used for the read-from-memory half and the last four states (S21, S22, S23, S24) for the write-to-memory half of the transfer.

IDLE CYCLE

When no channel is requesting service, the 8237 will enter the Idle cycle and perform "SI" states. In this cycle the 8237 will sample the DREQ lines every clock cycle to determine if any channel is requesting a DMA service. The device will also sample Chip Select (CS), looking for an attempt by the microprocessor to write or read the internal registers of the 8237. When CS is low and Hold Request (HRQ) is low ,the 8237 enters the Program Condition. The CPU can now establish, change or inspect the internal definition of the part by reading from or writing to the internal register. Address lines AO-A3 are inputs to the device and select which registers will be read or written. The lines Read and Write (IOR, IOW) are used to select and time reads or writes. Due to the number and size of the internal registers, an internal flip-flop is used to generate an additional bit of address. This bit is used to determine the upper or lower byte of the 16-bit Address and Word Count register. The flip-flop is reset by Master Clear or Reset. A separate software command can also reset this flip-flop.

Special software commands can be executed by the 8237 in the Program Condition. These commands are decoded as sets of addresses with the CS and IOW. The commands do not make use of the data bus. Instructions include Clear First/Last Flip-flop and Master Clear.

ACTIVE CYCLE

When the 8237 is in the Idle cycle and a channel requests a DMA service, the device will output an HRQ to the microprocessor and enter the Active cycle. It is in this cycle that the DMA service will take place, in one of three modes:

Single Transfer Mode - In Single Transfer mode the device is programmed to make one transfer only. The word count will be decremented and the address decremented or incremented following each transfer. When the word count goes to zero, a Terminal Count (TC) will cause an Autoinitialize if the channel has been programmed to do so.

Block Transfer Mode - In Block Transfer mode the device is activated by DREQ to continue making transfers during the service until a TC, caused by word count going to zero, is encountered. DREQ need only be held active until DACK becomes active. Again, an Autoinitialization will occur at the end of the service if the channel has been programmed for it.

Demand Transfer Mode - In Demand Transfer mode the device is programmed to continue making transfers until a TC is encountered or until DREQ goes inactive. Thus transfers may continue until the I/O device has had a chance to catch up, the DMA service is re-established by means of a DREQ. During the time between services when the microprocessor is allowed to operate, the intermediate values of address and word count are stored in the 8237 Current Address and Current Word Count registers. Only an EOP can cause an Autoinitialize at the end of the service. EOP is generated by Terminal Count (TC) .

TRANSFER TYPES

Each of the three active transfer modes can perform three different types of transfers. These are Read, Write and Verify. Write transfers move data form an I/O device to the memory by activating Memory Read (MEMR) and I/O write (IOW) .Verify transfers are pseudo transfers. The 8237 operates as in Read or

Write transfers generating addresses, and responding to EOP, etc. However, the memory and I/O control lines all remain inactive.

Memory-to-memory - To perform block moves of data from one memory address space to another with a minimum of program effort and time, the 8237 includes a memory-to-memory transfer feature. Programming a bit in the Command register selects channels 0 and 1 to operate as memory-to-memory transfer channels. The transfer is initiated by setting the software DREQ for channel 0. The 8237 requests a DMA service in the normal manner. After Hold Acknowledge (HLDA) is true ,the device , using eight-state transfer in Block Transfer mode, reads data from the memory. The channel 0 Current Address register is the source for the address used and is decremented or incremented in the normal manner. The data byte read from the memory is stored in the 8237 internal Temporary register. Channel 1 then writes the data from the Temporary register to memory using the address in its Current Address register and incrementing or decrementing it in the normal manner. The channel 1 Current Word Count is decremented. When the word count of Channel 1 goes to zero, a TC is generated causing an EOP output, terminating the service.

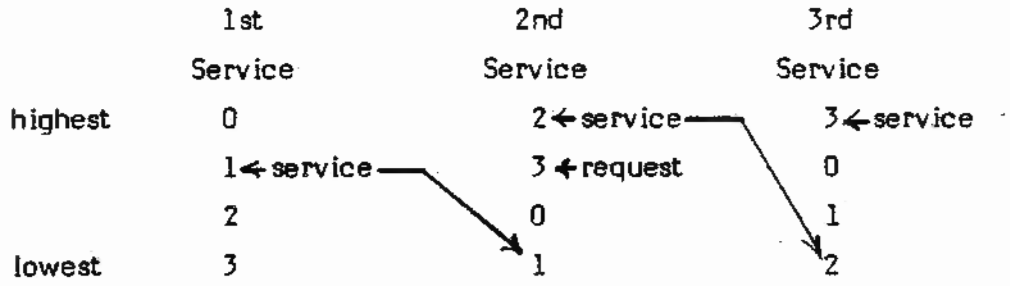
Channel 0 may be programmed to retain the same address for all transfers. This allows a single word to be written to a block memory.

The 8237 will respond to external EOP signals during memory-to-memory transfers. Data comparators in block search schemes may use this input to terminate the sevice when a match is found. The timing of memory-to-memory operations can be detected as an active Address Enable (AEN) with no DACK outputs.

Autoinitialize - By programming a bit in the Mode register, a channel may be set up as an Autoinitialize channel. During Autoinitialize initialization, the original values of the Current Address and Current Word Count register are automatically restored from the Base Address and Base Word Count registers of that channel following EOP. The base registers are loaded simultaneously with the current registers by the microprocessor and remain unchanged throughout the DMA service. The mask bit is not set when the channel is in Autoinitialize. Following Autoinitialize the channel is ready to perform another service without CPU intervention.

Priority - The 8237 has two types of priority encoding available as software selectable options. The first is Fixed Priority which fixes the channels in priority order based upon the descending value of their number. The channel with the lowest priority is 3 following by 2, 1 and the highest priority channel 0. After the recognition of any one channel for service, the other channels are prevented from interferring with that service until it is completed.

The second scheme is Rotating Priority. The last channel to get service becomes the lowest priority channel with the others rotating accordingly.



With Rotating Priority in a single chip DMA system, any device requesting service is guaranteed to be recognized after no more than three higher priority services have occurred. This prevents any one channel from monopolizing the system.

Compressed Timing - In order to achieve even greater throughput where system characteristics permit, the 8237 can compress the transfer time to two clocks. State S3 is used to extend the access time of the read pulse. By removing state S3, the read pulse width is made equal to the write pulse and a transfer consists only of state S2 to change the address and state S4 to perform the read/write. S1 states will still occur when A8-A15 need updating (see Address Generation).

Address Generation - In order to reduce pin count, the 8237 multiplexes the eight higher order address bits on the data lines. State S1 is used to output the higher order address bits to an external latch from which they may be placed on the address bus. The falling edge of Addres Strobe (ADSTB) is used to load these bits from the data lines to the latch. Address Enable (AEN) is used to enable the bits onto the address bus through a threestate enable. The lower order address bits are output by the 8237 directly.

During Block and Demand Transfer mode services, which include multiple transfers, the addresses generated will be sequential. For many transfers the data held in the external address latch will remain the same. This data need only change when a carry or borrow from A7 to A8 takes place in the normal sequence of addresses. To save time and speed transfers, the 8237 executes S1 states only when updating of A8-A15 in the latch is necessary. This means for long services, S1 states may occur only once every 256 transfers, a savings of 255 clock cycle for each 256 transfers.

REGISTER DESCRIPTION

Current Address Register - Each channel has a 16-bit Current Address register. This register holds the value of the address used during DMA transfers. The address is automatically incremented or decremented after each transfer and the intermediate values of the address are stored in the Current Address register during the transfer. This register is written or read by the microprocessor in successive 8-bit bytes. It may also be reinitialized by an Autoinitialize back to its orginal value. Autoinitialize takes place only after an EOP.

Current Word Register - Each channel has 16-bit Current Word Count register. This register holds the number of transfers to be performed. The word count is decremented after each transfer. The intermediate value of the word count is stored in the register during the transfer. When the value in the register goes to zero, a TC will be generated. This register is loaded or read in successive 8-bit bytes by the microprocessor in the Program Condition. Following the end of a DMA service it may also be reinitialized by an Autoinitialization back to its orginal value. Autoinitialize can occur only when an EOP occurs.

Base Address and Base Word Count Register - Each channel has a pair of Base Address and Base Word Count register. These 16-bit registers store the original value of their associated current registers. During Autoinitialize these values are used to restore the current registers to their original values. The base registers are written simultaneously with their corresponding current register in 8 bit- bytes in the Program Condition by the microprocessor. These registers cannot be read by the microprocessor.

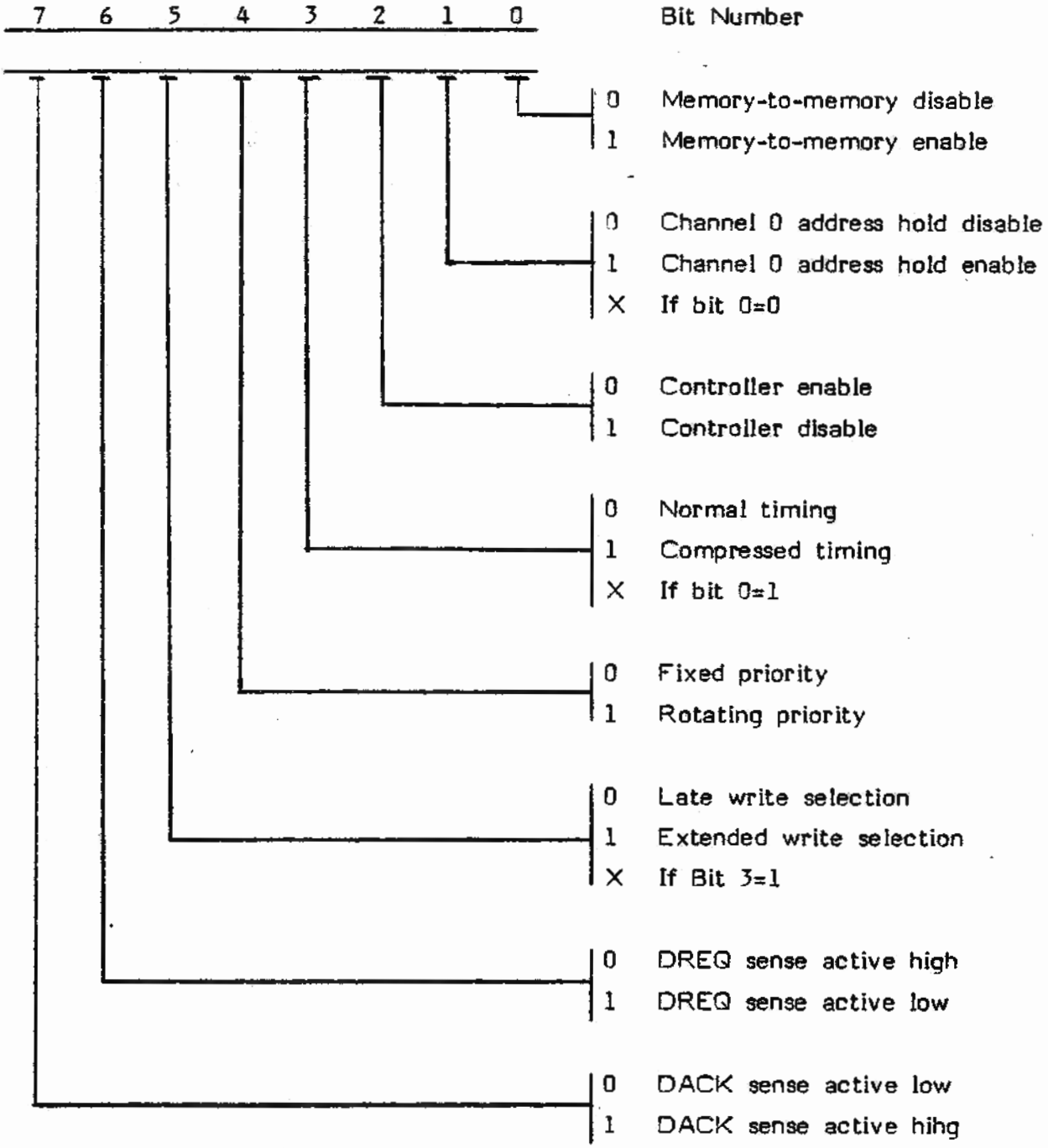
Command Register - This 8-bit register controls the operation of the 8237. It is programmed by the microprocessor in the Program Condition and is cleared by Reset. The following table lists the function of the command bits.

Mode Register - Each channel has a 6-bit Mode register associated with it. When the register is being written to by the microprocesor in the Program Condition, bits 0 and 1 determine which channel Mode register is to be written.

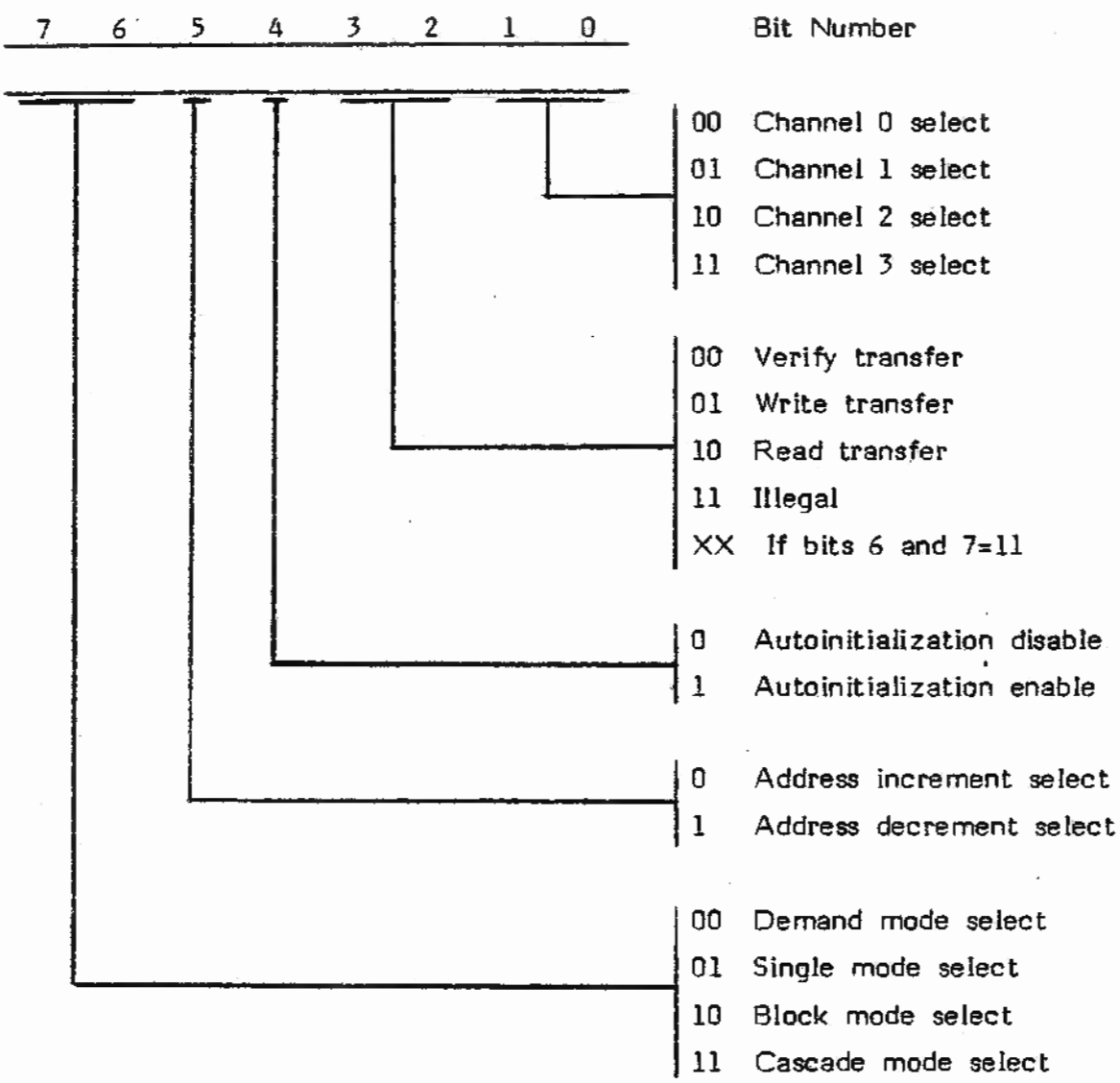
Request Register - The 8237 can respond to requests for DMA service which are initiated by software as well as by a DREQ. Each channel has a request bit associated with it in the 4-bit Request register. These are nonmaskable and subject to prioritization by the Priority Encoder network. Each register bit is set or reset separately under software control or is cleared upon generation of a TC or external EOP. The entire register is cleared by a Reset. To set or reset a bit, the software loads the proper form of the data word.

REGISTER BIT LEVEL DEFINITIONS

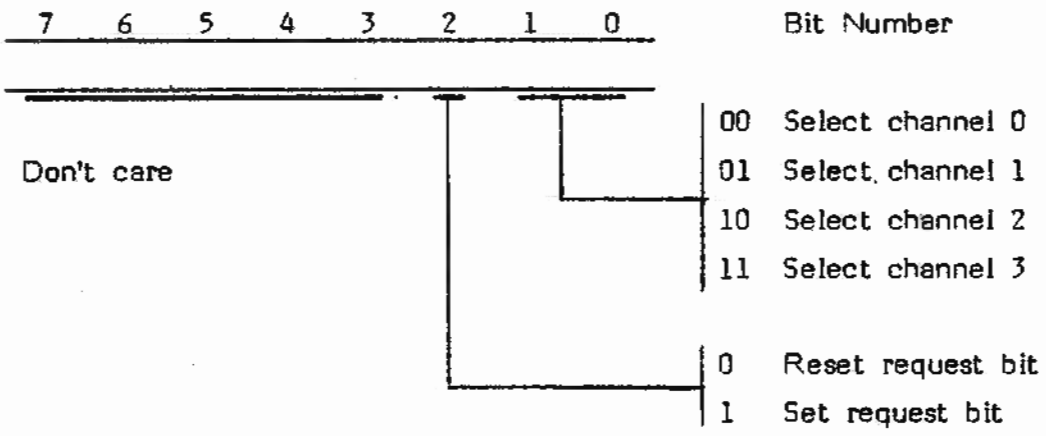
Command Register



Mode Register

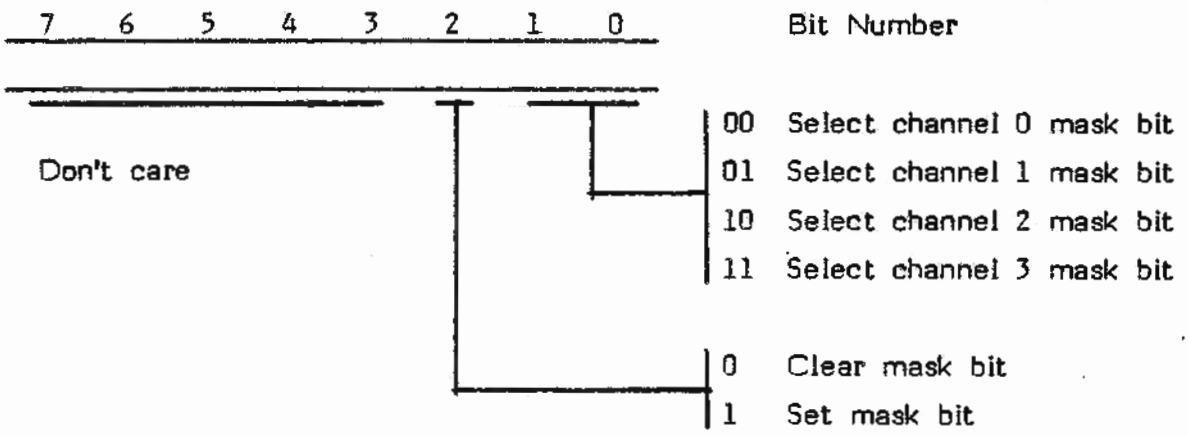


Request Register

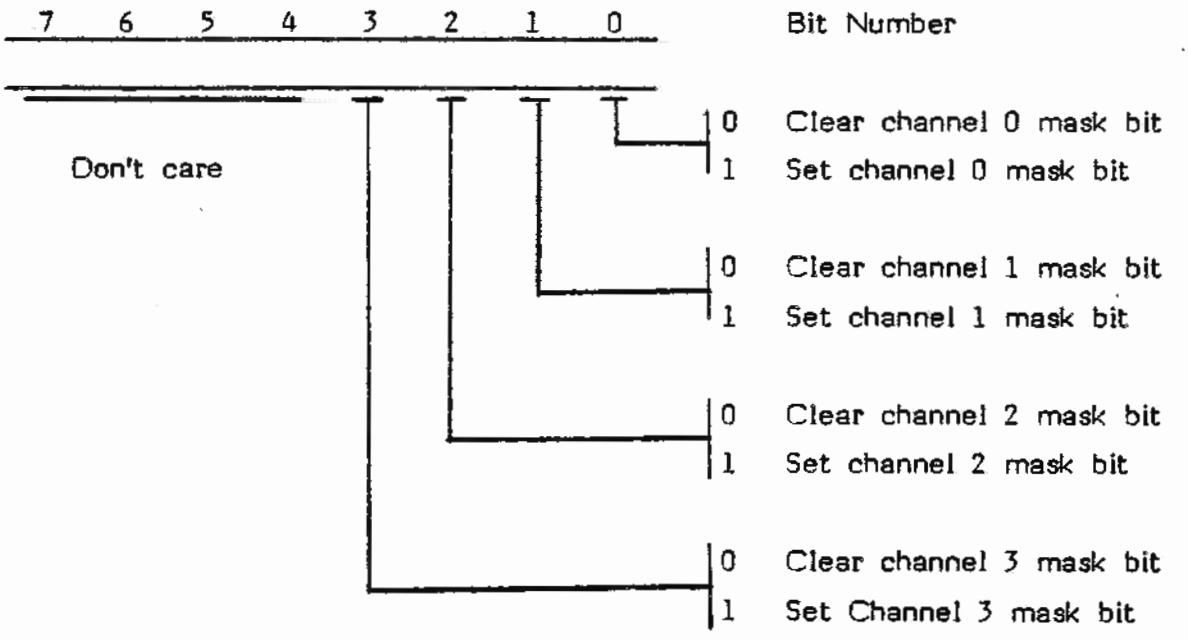


Software requests will be serviced only if the channel is in Block mode. When initiating a memory-to-memory transfer, the software request for channel 0 should be set.

Mask Register - Each channel has associated with it a mask bit which can be set to disable the incoming DREQ. Each mask bit is set when its associated channel produces an EOP if the channel is not programmed for Autoinitialize. Each bit of the 4-bit Mask register may also be set or cleared separately under software control. The entire register is also set by a Reset. This disables all DMA requests until a clear Mask register instruction allows them to occur. The instruction to separately set or clear the mask bits is similar in form to that used with the Request register. See instruction addressing.



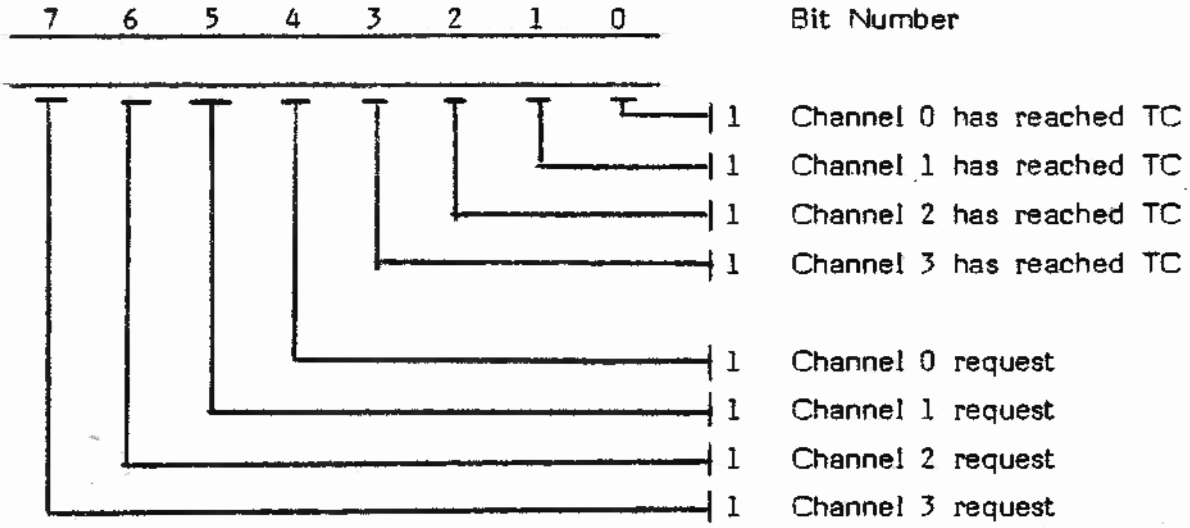
All four bits of the Mask register may also be written with a single command.



Register	Operation	R/W	Address
Command	Write	W	FFO8
Mode	Write	W	FFOB
Request	Write	W	FFO9
Mask	Set/Reset	W	FFOA
Mask	Write	W	FFOF
Temporary	Read	R	FFOD
Status	Read	R	FFO8

Definition of Register Codes

Status Register - The Status register is available to be read out of the 8237 by the microprocessor. It contains information about the status of the devices at this point. This information includes which channels have reached a terminal count and which channel have pending DMA requests. Bits 0-3 are set every time a TC is reached by that channel or an external EOP is applied. These bits are cleared upon Reset and on each Status Read. Bits 4-7 are set whenever their corresponding channels is requesting service.



Temporary Register - The Temporary register is used to hold data during memory-to-memory transfers. Following the completion of the transfers, the last word moved can be read by the microprocessor in the Program Condition. The Temporary register always contains the last byte transferred in the previous memory-to-memory operation, unless cleared by a Reset.

Software Commands - These are additional special software commands which can be executed in the Program Condition. They do not depend on any specific bit pattern on the data bus. The two software commands are:

Clear Fisrt/Last Flip-Flop: This command is executed prior to writing or reading new address or word count information to the 8237. This initializes the flip-flop to a known state so that subsequent accesses to register contents by the microprocessor will address upper and lower bytes in the correct sequence.

Master Clear: This software instruction has the same effect as the hardware Reset. The Command, Status, Request, Temporary, and Internal First/Last Flip-Flop registers are cleared and the Mask register is set. The 8237 will enter the Idle cycle.

ADDRESS	OPERATION	INTERN FLIP-FLOP	EXPLANATION	
FF00	WRITE	0	channel 0 Base and current address	A0-A7
	WRITE	1	channel 0 Base and current address	A8-A15
	READ	0	channel 0 Current address	A0-A7
	READ	1	channel 0 Current address	A8-A15
FF01	WRITE	0	channel 0 Base and current word count	W0-W7
	WRITE	1	channel 0 Base and current word count	W8-W15
	READ	0	channel 0 Current word count	W0-W7
	READ	1	channel 0 Current word count	W8-W15
FF02	WRITE	0	channel 1 Base and current addres	A0-A7
	WRITE	1	channel 1 Base and current address	A8-A15
	READ	0	channel 1 Current address	A0-A7
	READ	1	channel 1 Current address	A8-A15
FF03	WRITE	0	channel 1 Base and current word count	W0-W7
	WRITE	1	channel 1 Base and current word count	W8-W15
	READ	0	channel 1 Current word count	W0-W7
	READ	1	channel 1 Current word count	W8-W15
FF04	WRITE	0	channel 2 Base and current address	A0-A7
	WRITE	1	channel 2 Base and current address	A8-A15
	READ	0	channel 2 Current address	A0-A7
	READ	1	channel 2 Current address	A8-A15
FF05	WRITE	0	channel 2 Base and current word count	W0-W7
	WRITE	1	channel 2 Base and current word count	W8-W15
	READ	0	channel 2 Current word count	W0-W7
	READ	1	channel 2 Current word count	W8-W15
FF06	WRITE	0	channel 3 Base and current address	A0-A7
	WRITE	1	channel 3 Base and current address	A8-A15
	READ	0	channel 3 Current address	A0-A7
	READ	1	channel 3 Current address	A8-A15
FF07	WRITE	0	channel 3 Base and current word count	W0-W7
	WRITE	1	channel 3 Base and current word count	W8-W15
	READ	0	channel 3 Current word count	W0-W7
	READ	1	channel 3 Current word count	W8-W15
FF08	WRITE	x	Command register	
	READ	x	Status register	
FF09	WRITE	x	Request register	
	READ	x	ILLEGAL	
FF0A	WRITE	x	Single Mask Register Bit	
	READ	x	ILLEGAL	
FF0B	WRITE	x	Mode Register	
	READ	x	ILLEGAL	
FF0C	WRITE	x	Clear Byte pointer Flip Flop (INTERN FlipFlop above)	
	READ	x	ILLEGAL	
FF0D	WRITE	x	MASTER CLEAR	
	READ	x	Redtemporary register	
FF0E	WRITE	x	ILLEGAL	
	READ	x	ILLEGAL	
FF0F	WRITE	x	WRITE ALL MASK REGISTER BITS	
	READ	x	ILLEGAL	

x=NOT IN USE

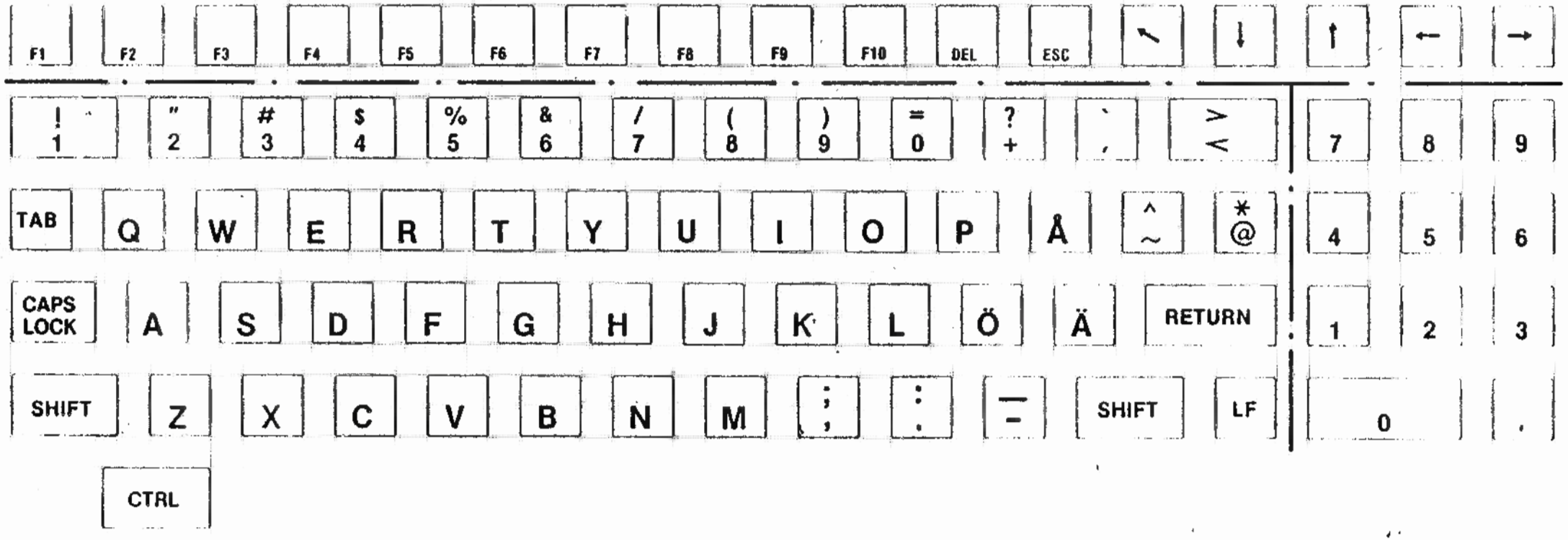
DMA COMMAND CODES

2.3 KEYBOARD INTERFACING

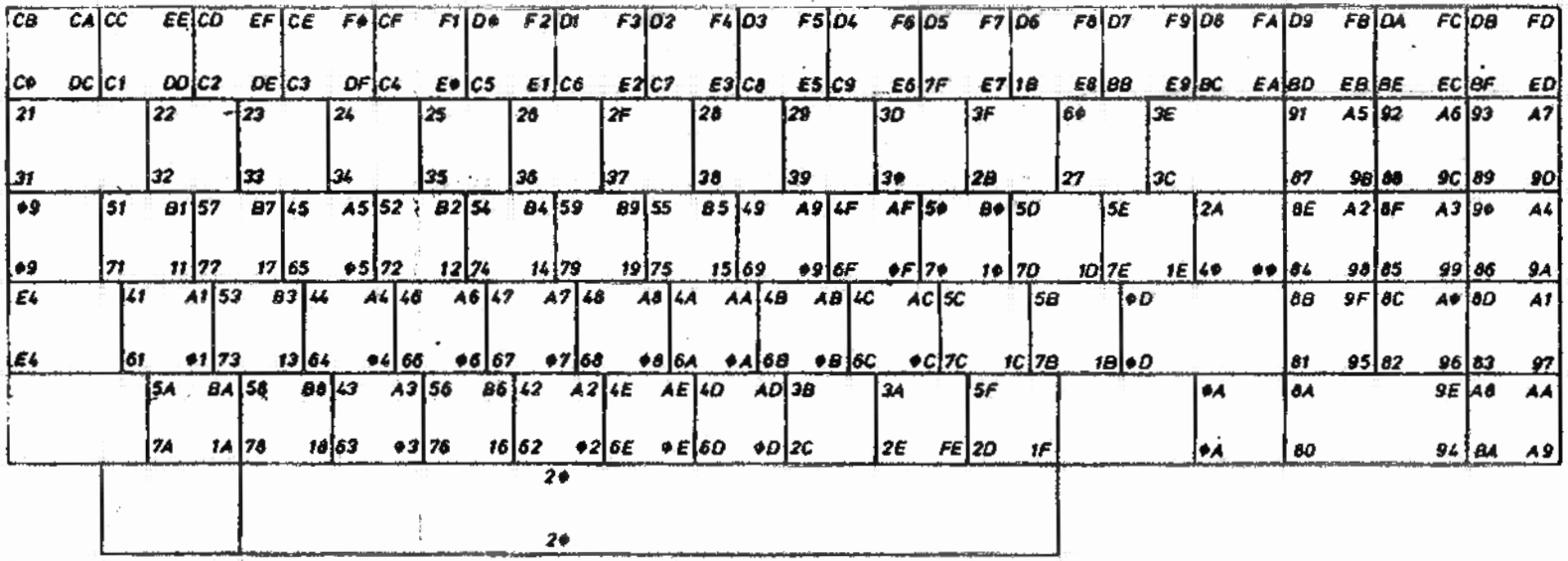
Keyboard interface in MicroMikko is parallel, 8-bit width.

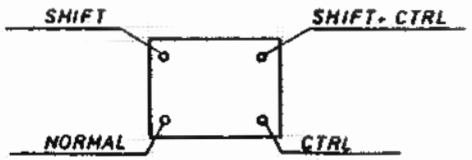
Keyboard logig generates an interrupt - RST 6.5 - request to CPU. This interrupt is hold active so long that CPU initiates an READ operation to device. Device is located into FF40H in I/O address space. Bell-feature in the keyboard is coupled into the SOD output of CPU, and beep-tone is given when SOD output is low.

Following pages illustrate keyboard layouts and corresponding scan codes.

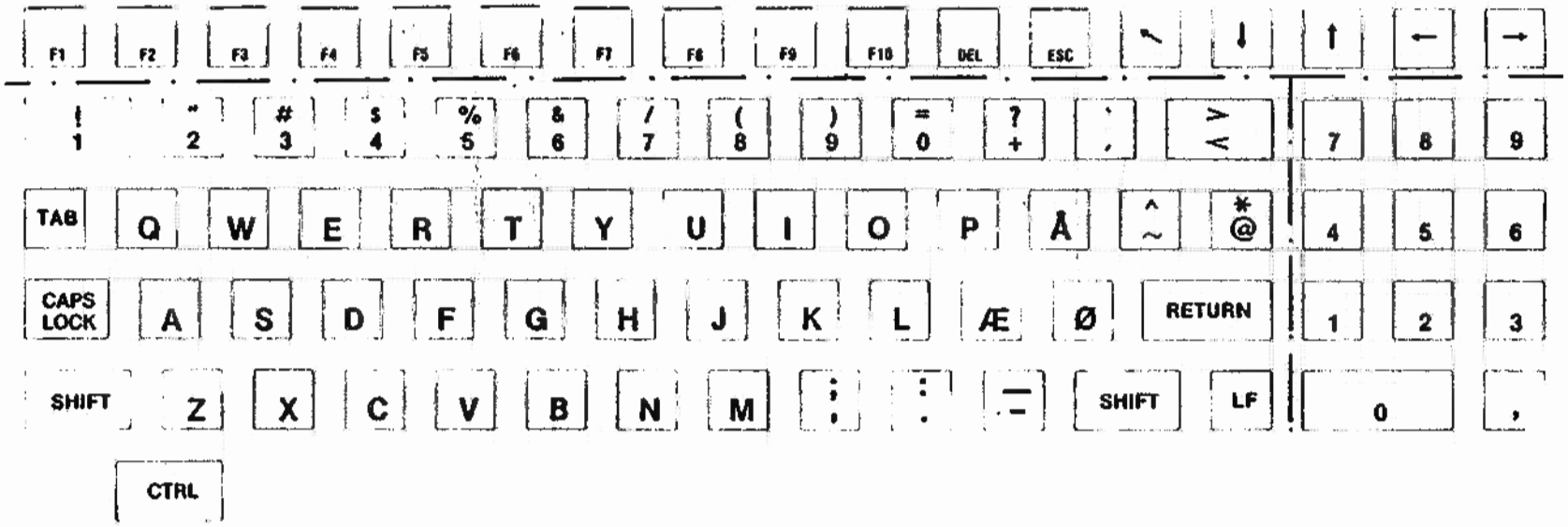


MicroMikko Keyboard SF/S

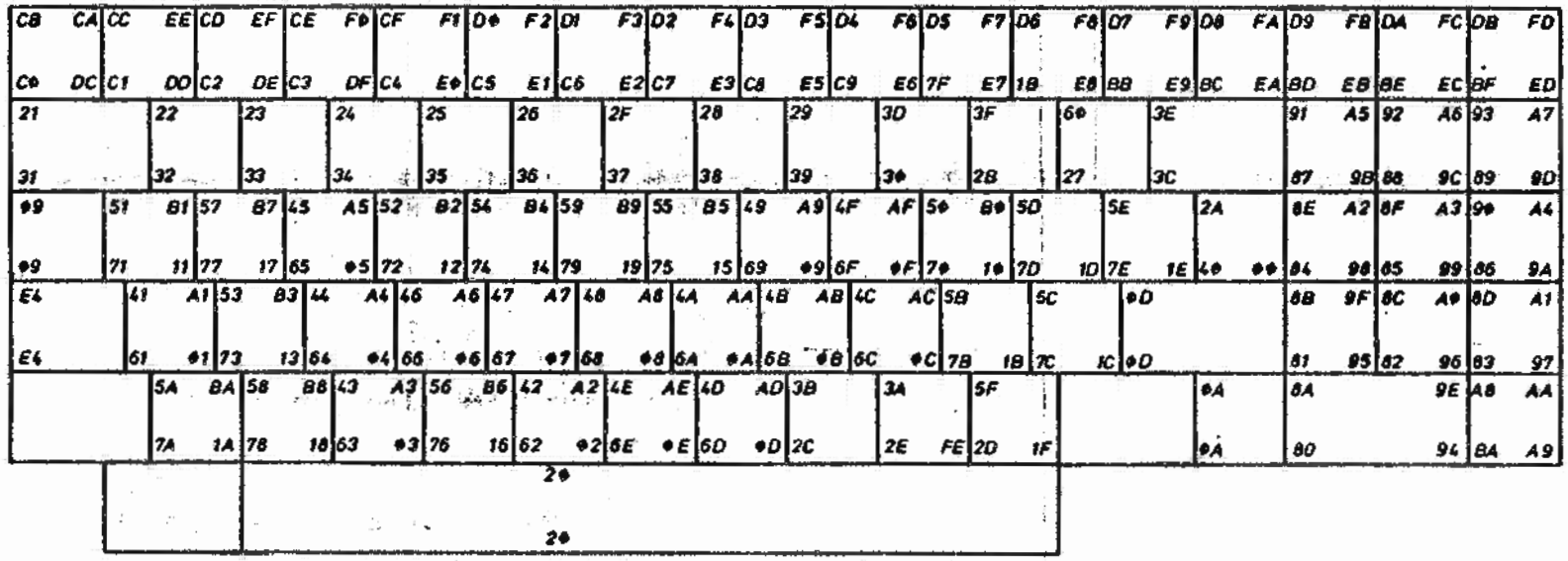


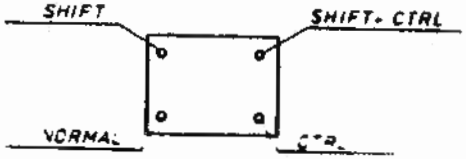


MicroMikko Keyboard Scan Codes SF/S

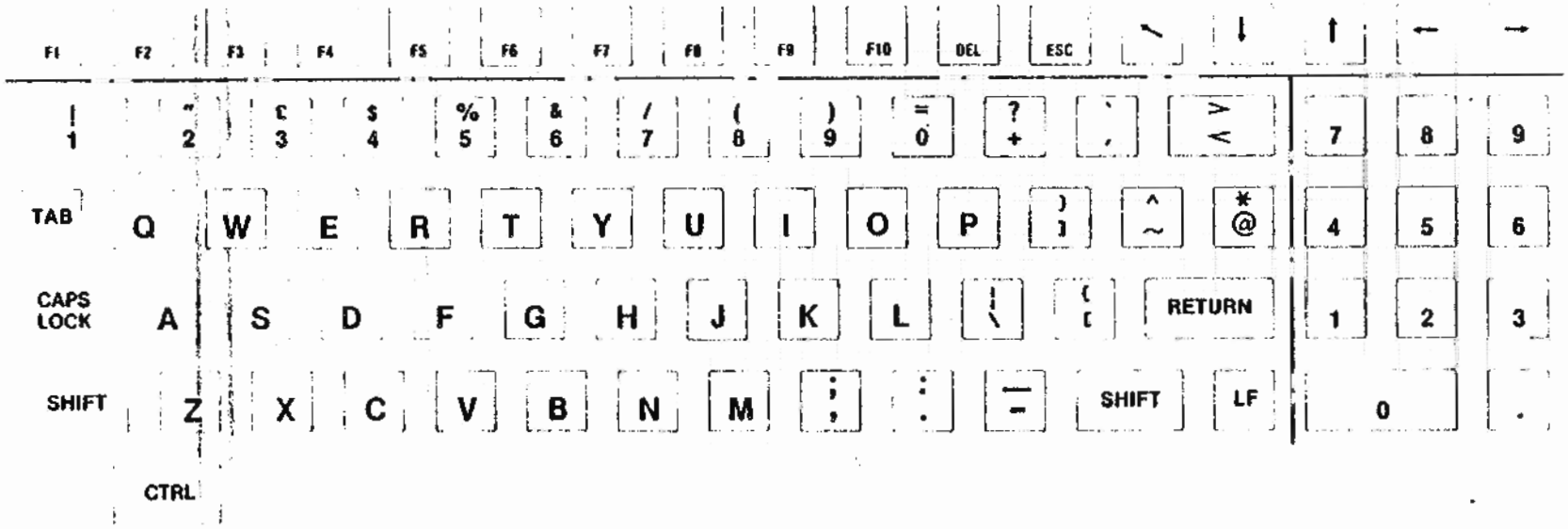


MicroMikko Keyboard N/DK

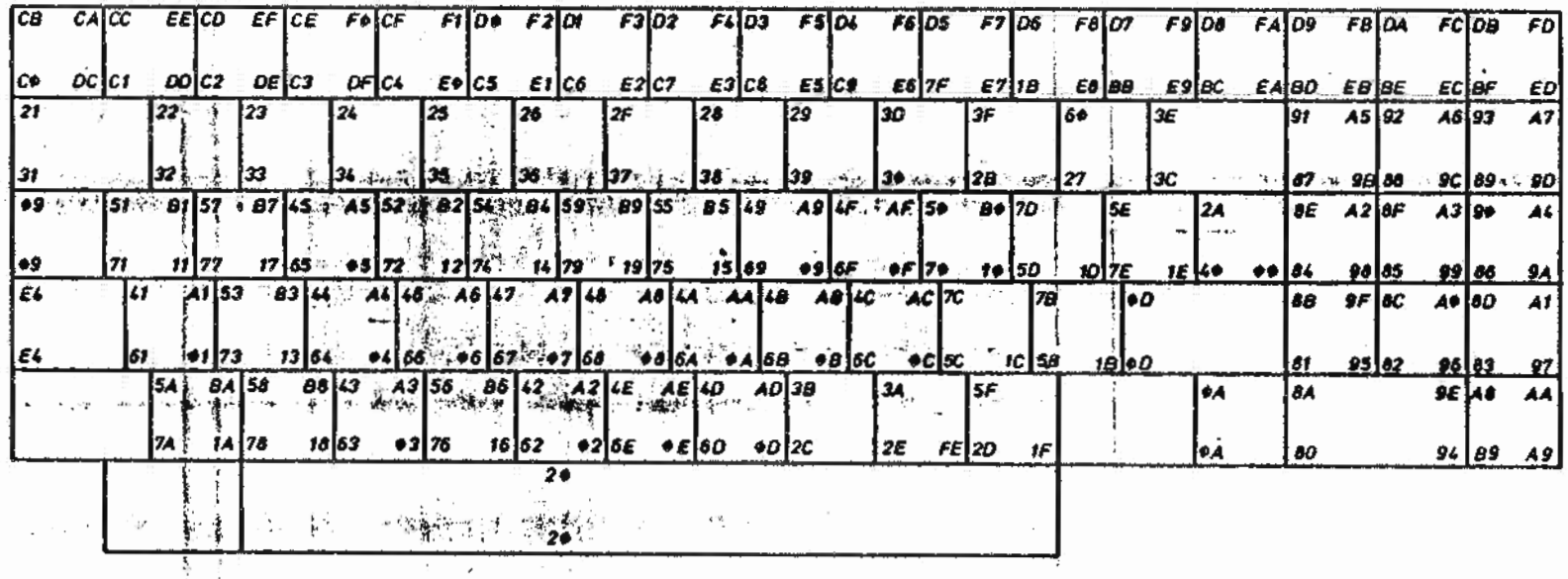


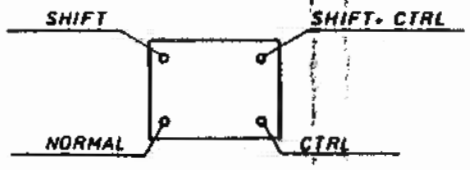


MicroMikko Keyboard Scan Codes N/DK

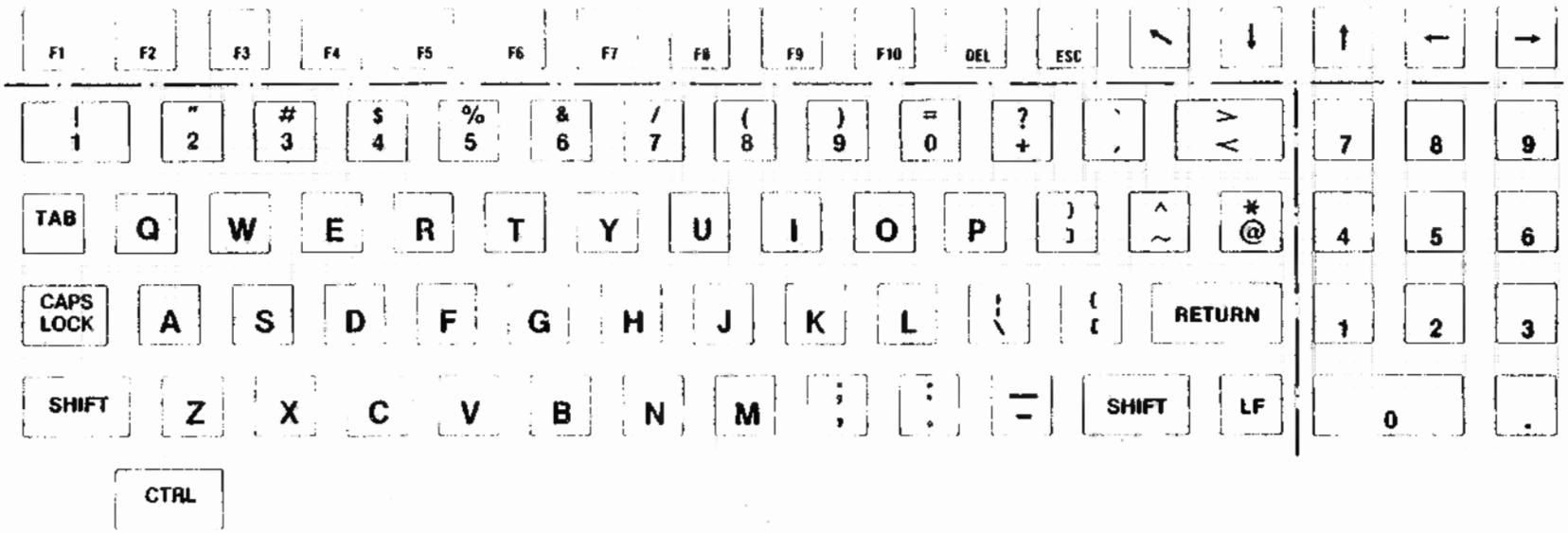


MicroMikko Keyboard GB

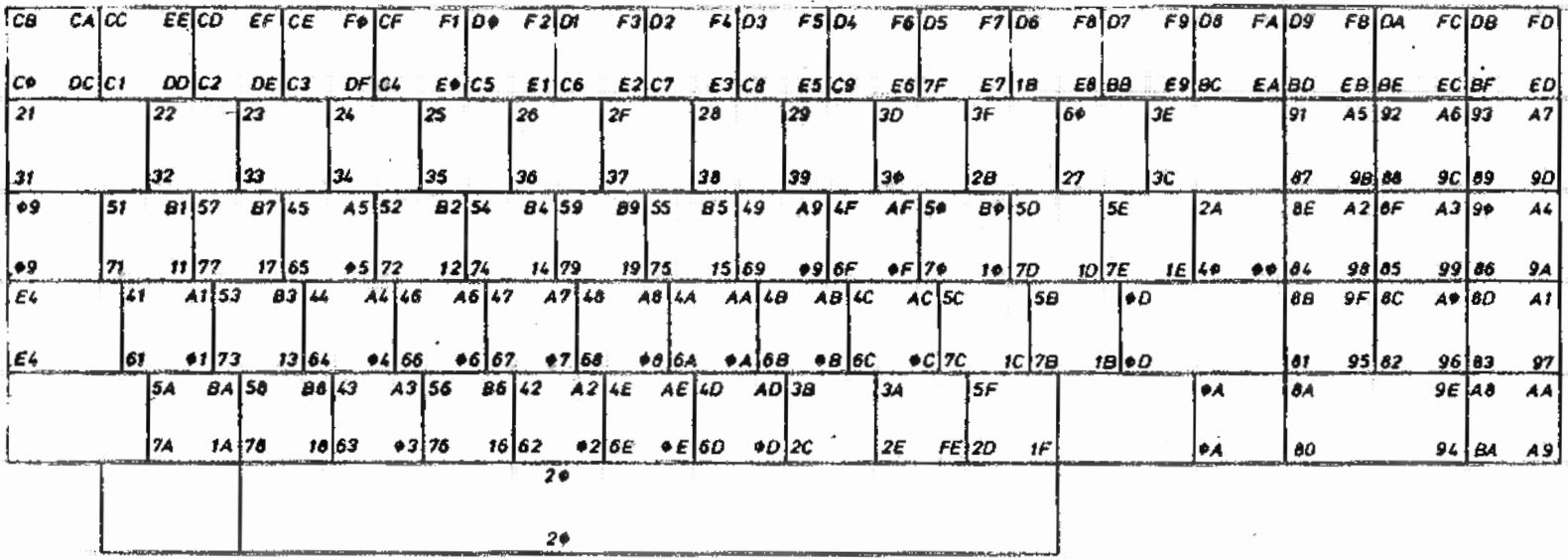


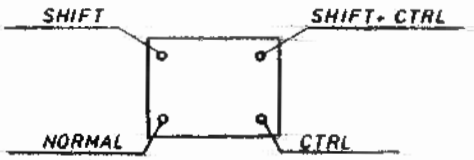


MicroMikko Keyboard Scan Codes GB

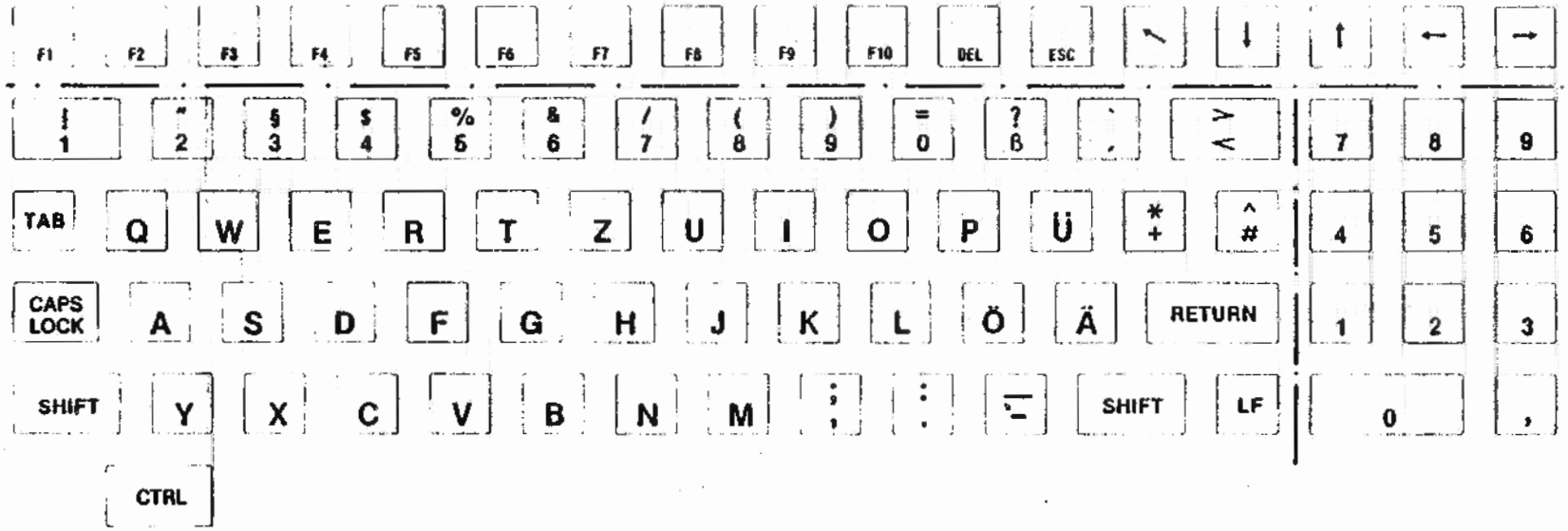


MicroMikko Keyboard USASCII

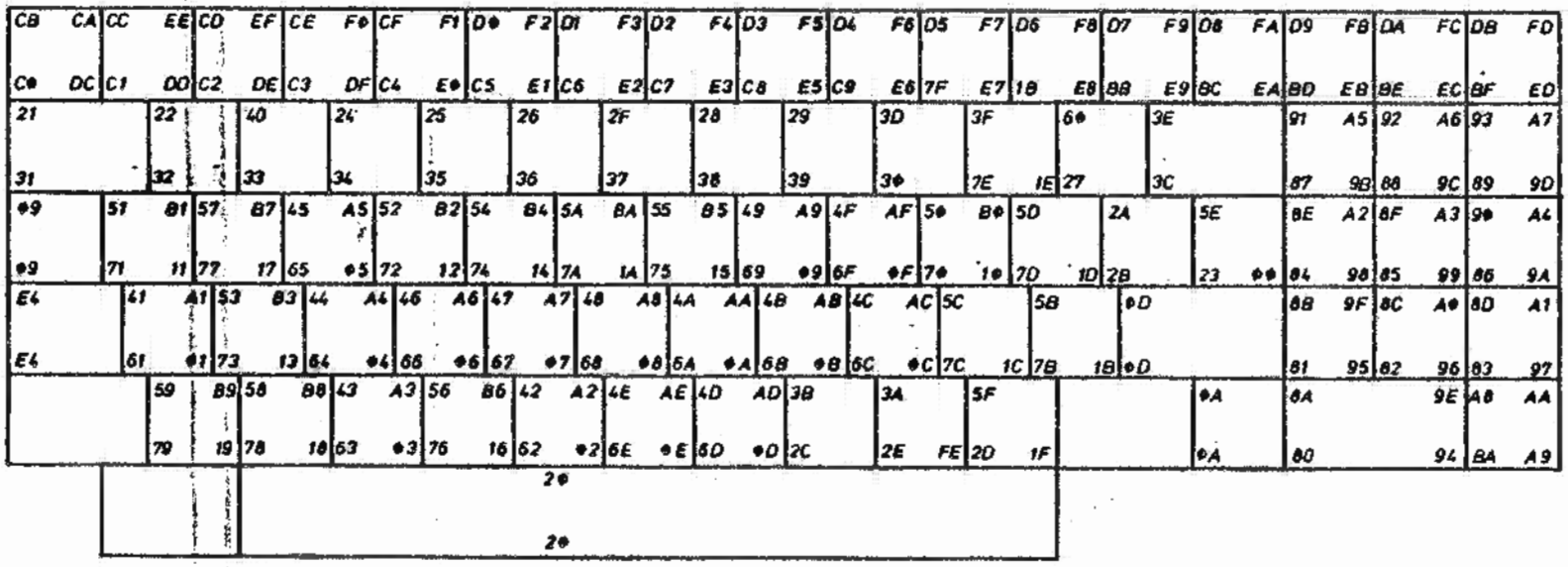




MicroMikko Keyboard Scan Codes USASCII



MicroMikko Keyboard D



MicroMikko Keyboard Scan Codes D

2.4 CRT CONTROL

CRT CONTROLLER INITIALIZATION

After that DMA is initialized, must also the CRT controller be initialized to communicate with DMA.

CRT controller is realized with Intels 8275. This is a programmable device, being able to handle display independent, without CPU intervention, with DMA controller after initialization. CRT controller provides character generator and all timing and control logic needed.

DMA INITIALIZATION

Programming DMA for display purposes.

DMA channel 0 is used for CRT-control. This channel must be programmed to transfer characters to be displayed to CRT-controller, and also to refresh Dynamic ROM. MicroMikko operating system uses following values.

Base and current address (FF00H) is set to F720H
Base and current wordcount (FF01H) is set to 7CFH (=2000 des)

Command register (SEE also DMA)

0 0 1 1 0 0 0 0	WRITE FF08

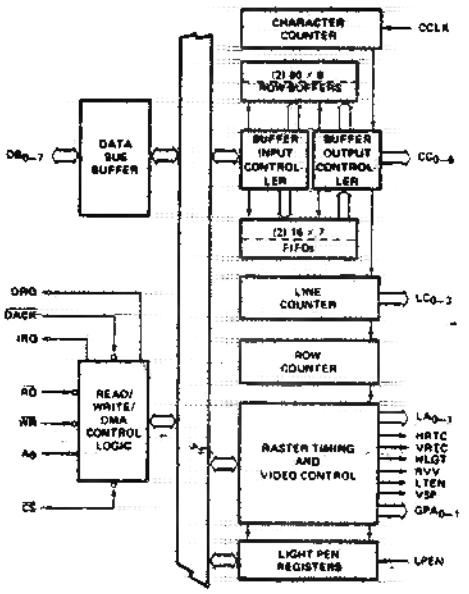
Mode Register

0 0 0 1 1 0 0 0	WRITE FF0B

Mask Register bit is reset

0 0 0 0 0 0 0 0	WRITE FF0A

FUNCTIONAL DESCRIPTION OF CRT CONTROLLER



This functional block accepts inputs from the System Control Bus and generates control signals for overall device operation. It contains the Command, Parameter, and Status Registers that store the various control formats for the device functional definition.

ADDRESS	R/W	REGISTER
FF20	R	PREG
FF20	W	PREG
FF21	R	SREG
FF21	W	CREG

ADDRESS	R/W	OPERATION
FF20	W	WRITE 8275 Parameter
FF20	R	READ 8275 Parameter
FF21	W	WRITE 8275 Command
FF21	R	READ 8275 Status

Character Counter

The Character Counter is a programmable counter that is used to determine the number of characters to be displayed per row and the length of horizontal retrace interval. It is driven by the CCLK (Character Clock) input, which should be a derivative of the external dot clock.

Line Counter

The Line Counter is a programmable counter that is used to determine the number of horizontal lines (Sweeps) per character row. Its outputs are used to address the external character generator ROM.

Row Counter

The Row Counter is a programmable counter that is used to determine the number of character rows to be displayed per frame and length of the vertical retrace interval.

Raster Timing an Video Controls

The Raster Timing circuitry controls the timing of the HRTC (Horizontal Retrace) and VRTC (Vertical Retrace) outputs. The Video Control circuitry controls the generation of LA_{0-1} (Line Attribute), HGLT (Highlight), RVV (Reverse Video), LTEN (Light Enable), VPS (Video Suppress), and GPA_{0-1} (General Purpose Attribute) outputs.

Row Buffers

The Row Buffers are two 80 character buffers. They are filled from the microcomputer system memory with the character codes to be displayed. While one row buffer is displaying a row of characters, the other is being filled with the next row of characters.

FIFOs

There are two 16 character FIFOs in the 8275. They are used to provide extra row buffer length in the Transparent Attribute Mode (see Detailed Operation section).

Buffer Input/Output Controllers

The Buffer Input/Output Controllers decode the characters being placed in the row buffers. If the character is a character attribute, field attribute or special code, these controllers control the appropriate action. (Examples: An " End of Screen-Stop DMA" special code will cause the Buffer Input Controller to stop further DMA requests. A "Highlight" field attribute will cause the Buffer Output Controller to active the HGLT output.)

General Systems Operational Description

The 8275 provides a "window" into the microcomputer system memory.

Display characters are retrieved from memory and displayed on a row by row basis. The 8275 has two row buffers. While one row buffer is being used for display, the other is being filled with the next row of characters to be displayed. The number of display characters per row and the number of characters rows per frame are software programmable. (See Programming Section.)

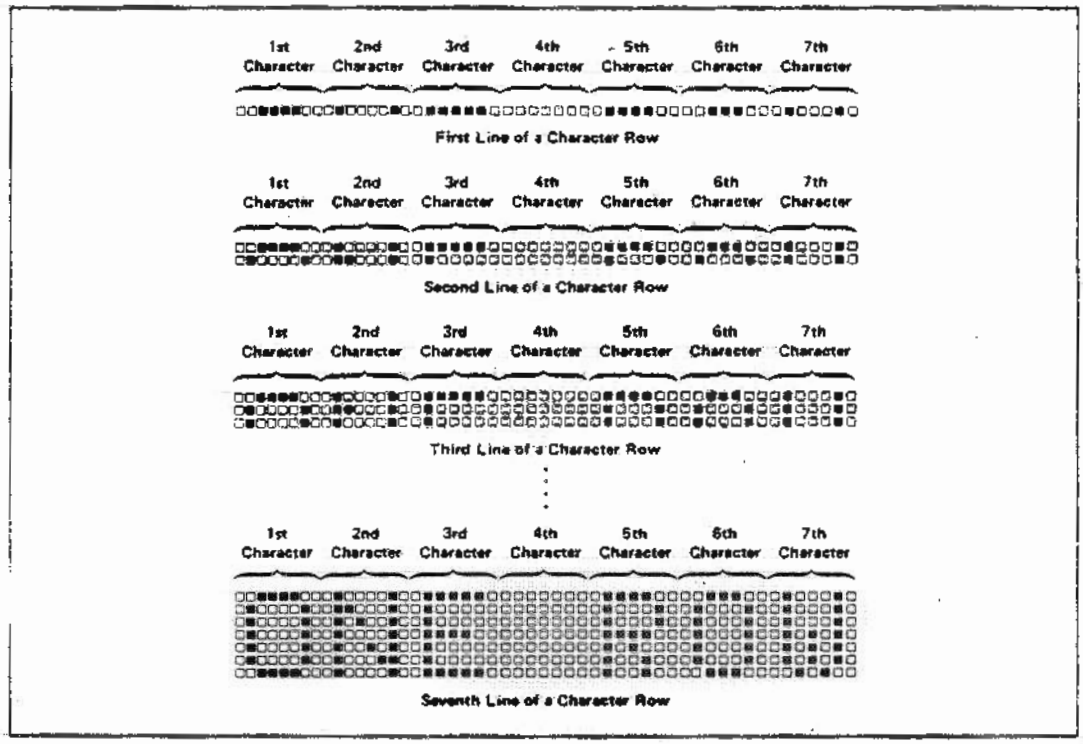
The 8275 requests DMA to fill the row buffer that is not being used for display. DMA burst length and spacing is programmable. (See Programming Section.)

The 8275 displays character rows one line at a time. The number of lines per character row, the underline position, and blanking of top and bottom lines are programmable. (See Programming Section.)

The 8275 provides special Control Codes which can be used to minimize DMA or software overhead. It also provides Visual Attribute Codes to cause special action or symbols on the screen without the use of the character generator (see Visual Attributes Section).

The 8275 also controls raster timing. This is done by generating Horizontal Retrace (HRTC) an Vertical Retrace (VRTC signals. The timing of these signals is programmable.

The 8275 can generate a cursor. Cursor location and format are programmable. (See Programming Section.)



Display Row Buffering

Before the start of a frame, the 8275 requests DMA and one buffer is filled with characters.

When the first horizontal sweep is started, character codes are output to the character generator from the row buffer just filled. Simultaneously, DMA begins filling the other row buffer with the next row of characters.

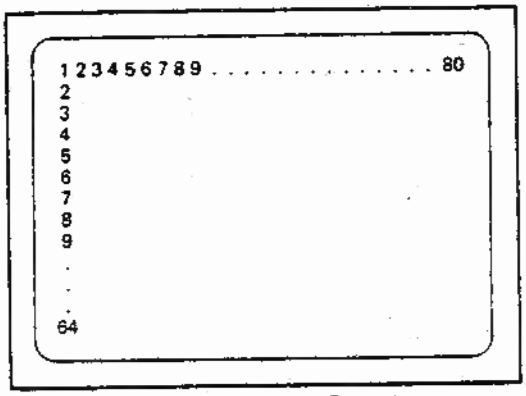
After all the lines of the character row are scanned, the roles of the two row buffers are reversed and the same procedure is followed for the next row.

This is repeated until all of the character rows are displayed.

DISPLAY FORMAT

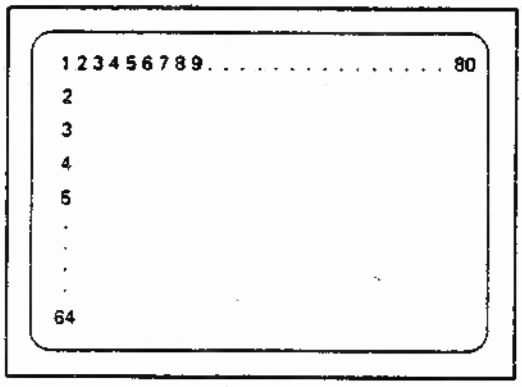
Screen Format

The 8275 can be programmed to generate from 1 to 80 characters per row, and from 1 to 64 rows per frame.



Screen Format

The 8275 can also be programmed to blank alternate rows. In this mode, the first row is displayed, the second blanked, the third displayed, tec. DMA is not requested for the blanked rows.



Blank Alternate Rows Mode

Row Format

The 8275 is designed to hold the line count stable while outputting the appropriate character codes during each horizontal sweep. The line count is incremented during horizontal retrace and the whole row of character codes are output again during the next sweep. This is continued until the whole character row is displayed.

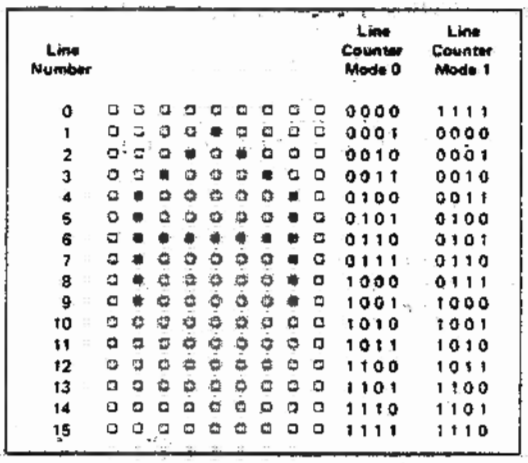
The number of lines (horizontal sweeps) per character row is programmable from 1 to 16.

The output of the line counter can be programmed to be in one of two modes.

In mode 0, the output of the line counter is the same as the line number.

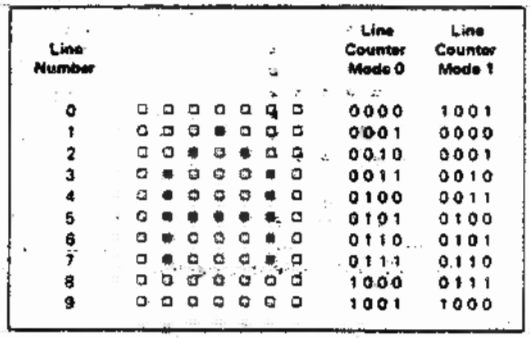
In mode 1, the line counter is offset by one from the line number.

Note: In mode 1, while the first line (line number 0) is being displayed, the last count is output by the line counter (see examples).



Line Number	Line Counter Mode 0	Line Counter Mode 1
0	0000	1111
1	0001	0000
2	0010	0001
3	0011	0010
4	0100	0011
5	0101	0100
6	0110	0101
7	0111	0110
8	1000	0111
9	1001	1000
10	1010	1001
11	1011	1010
12	1100	1011
13	1101	1100
14	1110	1101
15	1111	1110

Example of a 16-Line Format



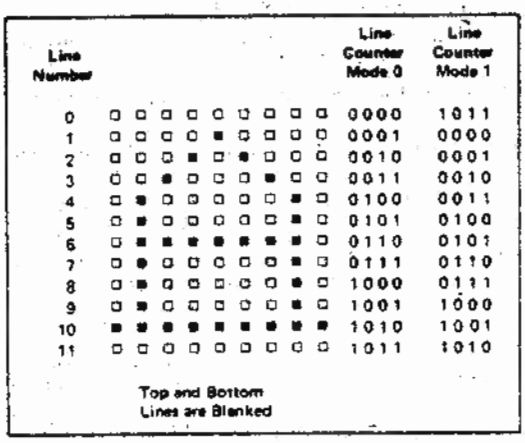
Line Number	Line Counter Mode 0	Line Counter Mode 1
0	0000	1001
1	0001	0000
2	0010	0001
3	0011	0010
4	0100	0011
5	0101	0100
6	0110	0101
7	0111	0110
8	1000	0111
9	1001	1000

Example of a 10-Line Format

Mode 0 is useful for character generators that leave address zero blank and start at address 1. Mode 1 is useful for character generators which start at address zero.

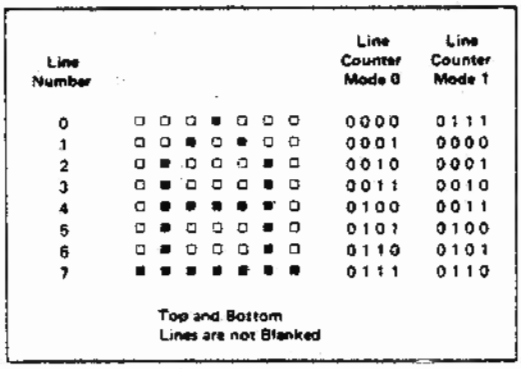
Underline placement is also programmable (from line number 0 to 15). This is independent of the line counter mode.

If the line number of the underline is greater than 7 (line number MSB = 1), then the top and bottom lines will be blanked.



Underline in Line Number 10

If the line number of the underline is less than or equal to 7 (line number MSB = 0), then the top and bottom lines will not be blanked.



Underline in Line Num 7

If the line number of the underline is greater than the maximum number of lines, the underline will not appear.

Raster Timing

The character counter is driven by the character clock input (CCLK). It counts out the characters being displayed (programmable from 1 to 80). It then causes the line counter to increment, and it starts counting out the horizontal retrace interval (programmable from 2 to 32). This is constantly repeated.

The line counter is driven by the character counter. It is used to generate the line address outputs for the character generator. After it counts all of the lines in a character row (programmable from 1 to 16), it increments the row counter, and starts over again.

The row counter is an internal counter driven by the line counter. It controls the functions of the row buffers and counts the number of character rows displayed.

After the row counter counts all of the rows in a frame (programmable from 1 to 64), it starts counting out the vertical retrace interval (programmable from 1 to 4).

The Video Suppression Output is active during horizontal and vertical retrace intervals.

DMA Timing

The 8275 can be programmed to request burst DMA transfers of 1 to 8 characters. The interval between bursts is also programmable (from 0 to 55 character clock periods $\pm$ 1).

The first DMA request of the frame occurs one row time before the end of vertical retrace. DMA reguests continue as programmed, until the row buffer is filled. If the row buffer is filled in the middle of a burst, the 8275 terminates the burst and resets the burst counter. No more DMA requests will occur until the beginning of the next row. At that time, DMA requests are activated as programmed until the other buffer is filled.

The first DMA request for a row will start at the first character clock of the preceding row. If the burst mode is used, the first DMA request may occur a number of character clocks later. This number is equal to the programmed burst space.

If for any reason, there is a DMA underrun, a flag in the status word will be set.

The DMA controller is typically initialized for the next frame at the end of the current frame.

Interrupt Timing

The 8275 can be programmed to generate an interrupt request at the end of each frame. This can be used to reinitialize the DMA controller. If the 8275 interrupt enable flag is set, an interrupt request will occur at the beginning of the last display row. This features is not used in MicroMikko.

VISUAL ATTRIBUTES AND SPECIAL CODES

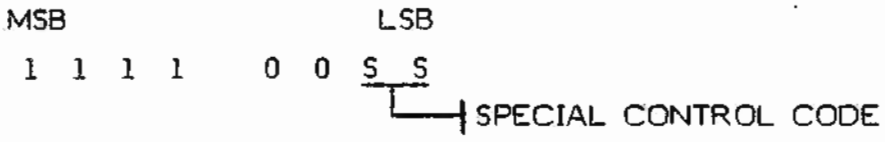
The characters processed by the 8275 are 8 bit quantities. The character code outputs provide the character generator with 7 bits of address. The Most Significant Bit is the extra bit and it is used to determine if it is a normal display character (MSB = 0), or if it is a Visual Attribute or Special Code (MSB = 1).

There are two types of Visual Attribute Codes. They are Character Attributes and Field Attribute Codes. Character Attribute codes are usually not used in MicroMikko.

Special Codes

Four special codes are available to help reduce memory, software, or DMA overhead.

Special Control Character



S	S	FUNCTION
0	0	End of Row
0	1	End of Row-Stop DMA
1	0	End of Screen
1	1	End of Screen-Stop DMA

The End of Row Code (00) activates VSP and holds it to the end of the line.

The End of Row-Stop DMA Code (01) causes the DMA Control Logic to stop DMA for the rest of the row when it is written into the Row Buffer. It affects the display in the same way as the End of Row Code (00).

The End of Screen Code (10) activates VSP and holds it to the end of the frame.

The End of Screen-Stop DMA Code (11) causes the DMA Control Logic to stop DMA for the rest of the frame when it is written into the Row Buffer. It affects the display in the same way as the End of Screen Code (10).

If the Stop DMA feature is not used , all characters after an End of Row character are ignored, except for the End of Screen character, which operates normally. All characters after an End of Screen character are ignored.

Note: If a Stop DMA character is not the last character in a burst or row, DMA is not stopped until after the next character is read. In this situation, a dummy character must be placed in memory after the Stop DMA character.

Field Attributes

The field attributes are control codes which affect the visual characteristics for a field of characters, starting at the character following the code up to, and including, the character which precedes the next field attribute code, or up to the end of the frame. The field attribute are reset during the vertical retrace interval.

There are six attributes:

1. Blink - Characters following the code are caused to blink by activating the Video Suppression output (VSP). The blink frequency is 1,6 hz

2. Low Brightness - Characters following the code are caused to be low lighted.

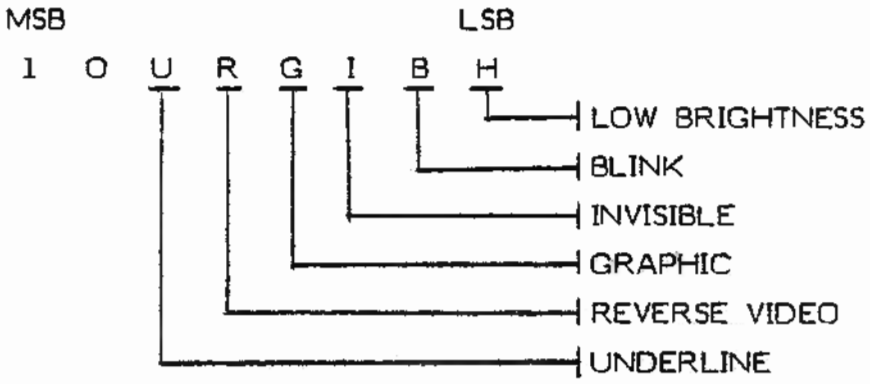
3. Reverse Video - Characters following the code are caused to appear with reverse video by activating the Reverse Video output.

4. Underline - Characters following the code are caused to be underlined by activating the Light Enable output.

5. Invisible - Characters following the code are invisible i.e. they are not displayed.

6. Graphic - Characters following the code are taken from the upper part of character generator offering High resolution graphic with limited character set.

Field Attribute Code

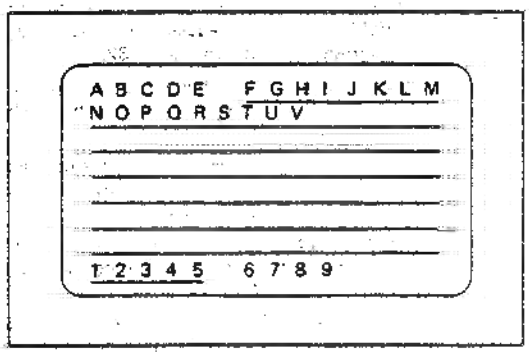


H = 1 FOR LOW BRIGHTNESS
B = 1 FOR BLINKING
R = 1 FOR REVERSE VIDEO
U = 1 FOR UNDERLINE
I = 1 FOR INVISIBLE
G = 1 FOR HIGH RESOLUTION GRAPHIC

More than one attribute can be enabled at the same time. If the blinking and reverse video attributes are enabled simultaneously, only the reversed characters will blink.

The 8275 can be programmed to provide invisible field attribute characters.

When the 8275 is programmed in the visible field attribute mode, all field attributes will occupy a position on the screen. They will appear as blanks caused by activation of the Video Suppression output (VSP). The chosen visual attributes are activated after this blanked character.



Example of the Visible Field Attribute Mode (Underline Attribute)

If the 8275 is programmed in the invisible field attribute mode, the 8275 FIFO is activated.

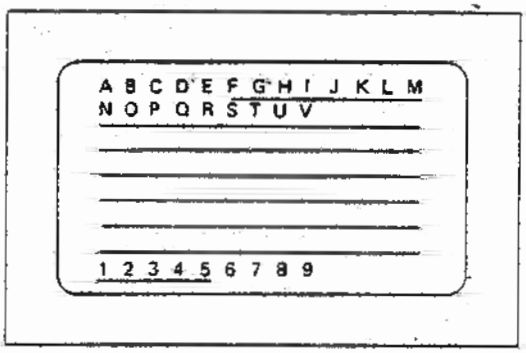
Each row buffer has a corresponding FIFO. These FIFOs are 16 characters by 7 bits in size.

When a field attribute is placed in the row buffer during DMA, the buffer input controller recognizes it and places the next character in the proper FIFO.

When a field attribute is placed in the Buffer Output Controller during display, it causes the controller to immediately put a character from the FIFO on the Character Code outputs. The chosen Visual Attributes are also activated.

Since the FIFO is 16 characters long, no more than 16 field attribute characters may be used per line in this mode. If more are used, a bit in the status word is set and the first characters in the FIFO are written over and lost.

Note: Since the FIFO is 7 bits wide, the MSB of any characters put in it are stripped off. Therefore, a Visual Attribute or Special Code must not immediately follow a field attribute code. If this situation does occur, the Visual Attribute or Special Code will be treated as a normal display character.



Example of the Invisible Field Attribute Mode (Underline Attribute)

MicroMikko is normally programmed to Visible Field Attribute Mode.

Cursor Timing

The cursor location is determined by a cursor row register and a character position register which are loaded by command to the controller. The cursor can be programmed to appear on the display as:

1. a blinking underline
2. a blinking reverse video block
3. a non-blinking underline
4. a non-blinking reverse video block

The cursor blinking frequency is equal to the screen refresh frequency divided by 16 = 3,2 hz.

If a non-blinking reverse video cursor appears in a non-blinking reverse video field, the cursor will appear as a normal video block.

If a non-blinking underline cursor appears in a non-blinking underline field, the cursor will not be visible.

DEVICE PROGRAMMING

The 8275 has two programming register, the Command Register (CREG) and the Parameter Register (PREG). It also has a Status Register (SREG). The Command Register can only be written into and the Status Register can only be read from. They are addressed as follows:

A_0	OPERATION	REGISTER
FF20H	Read	PREG
FF20H	Write	PREG
FF21H	Read	SREG
FF21H	Write	CREG

The 8275 expects to receive a command and a sequence of 0 to 4 parameters, depending on the command. If the proper number of parameter bytes are not received before another command is given, a status flag is set, indicating an improper command.

Instruction Set

The 8275 instruction set consists of 8 commands.

COMMAND	NO. OF PARAMETER BYTES
Reset	4
Start Display	0
Stop Display	0
Read Light Pen	2
Load Cursor	2
Enable Interrupt	0
Disable Interrupt	0
Preset Counters	0

In addition, the status of the 8275 (SREG) can be read by the CPU at any time.

Reset Command

	OPERATION	DESCRIPTION	DATA BUS MSB							LSB
Command	Write FF21H	Reset Command	0	0	0	0	0	0	0	0
Parameters	Write FF20H	Screen Comp Byte 1	S	H	H	H	H	H	H	H
	Write FF20H	Screen Comp Byte 2	V	V	R	R	R	R	R	R
	Write FF20H	Screen Comp Byte 3	U	U	U	U	L	L	L	L
	Write FF20H	Screen Comp Byte 4	M	F	C	C	Z	Z	Z	Z

Action - After the reset command is written, DMA requests stop, 8275 interrupts are disabled, and the VSP output is used to blank the screen. HRTC and VRTC continue to run. HRTC and VRTC timing are random on power-up.

As parameters are written, the screen composition is defined.

Parameter - S Spaced Rows

S	FUNCTIONS
0	Normal Rows
1	Spaced Rows

Parameter - HHHHHHH Horizontal Characters/ Row

H	H	H	H	H	H	H	NO. OF CHARACTERS PER ROW
0	0	0	0	0	0	0	1
0	0	0	0	0	0	1	2
0	0	0	0	0	1	0	3
			.				.
			.				.
			.				.
1	0	0	1	1	1	1	80

1	0	1	0	0	0	0	Undefined
			•				•
			•				•
			•				•
1	1	1	1	1	1	1	Undefined

Parameter - VV Vertical Retrace Row Count

V	V	NO. OF ROW COUNTS PER VRTC
0	0	1
0	1	2
1	0	3
1	1	4

Parameter - RRRRRR Vertical Rows/Frame

R	R	R	R	R	R	NO. OF ROW/FRAME
0	0	0	0	0	0	1
0	0	0	0	0	1	2
0	0	0	0	1	0	3
			•			•
			•			•
			•			•
1	1	1	1	1	1	64

Parameter - UUUU Underline Placement

U	U	U	U	LINE NUMBER OF UNDERLINE
0	0	0	0	1
0	0	0	1	2
0	0	1	0	3
		•		•
		•		•
		•		•

1	1	1	1	16

Parameter - LLLL Number of Lines per Character Row

L	L	L	L	NO. OF LINES/ROW
0	0	0	0	1
0	0	0	1	2
0	0	1	0	3
		.		.
		.		.
		.		.
1	1	1	1	16

Parameter - M Line Counter Mode

M	LINE COUNTER MODE
0	Mode 0 (Non-Offset)
1	Mode 1 (Offset by 1 Count)

Parameter - F Field Attribute Mode

F	FIELD ATTRIBUTE MODE
0	Transparent
1	Non- Transparent

Parameter - CC Cursor Format

C	C	CURSOR FORMAT
0	0	Blinking reverse video block
0	1	Blinking underline
1	0	Nonblinking reverse video block
1	1	Nonblinking underline

Parameter - ZZZZ Horizontal Retrace Count

Z Z Z Z	NO. OF CHARACTER COUNTS PER HRTC
0 0 0 0	2
0 0 0 1	4
0 0 1 0	6
.	.
.	.
.	.
1 1 1 1	32

Note: uuuu MSB determines blanking of top and bottom lines (1 = blanked, 0 = not blanked).

Start Display Command:

			DATA BUS							
	OPERATION	DESCRIPTION	MSB							LSB
Command	Write FF21H	Start Display	0	0	1	S	S	S	B	B
No parameters										

SSS BURST SPACE CODE

S S S	NO. OF CHARACTER CLOCKS BETWEEN DMA REQUESTS
0 0 0	0
0 0 1	7
0 1 0	15
0 1 1	23
1 0 0	31
1 0 1	39
1 1 0	47
1 1 1	55

BB BURST COUNT CODE

B	B	NO. OF DMA CYCLES PER BURST
0	0	1
0	1	2
1	0	4
1	1	8

Action - 8275 interrupts are enabled, DMA requests begin, video is enabled, Interrupt Enable and Video Enable status flags are set.

Stop Display Command:

	OPERATION	DESCRIPTION	DATA BUS MSB							LSB
Command	Write FF21H	Stop Display	0	1	0	0	0	0	0	0
No parameters										

Action - Disables video, interrupts remain enabled, HRTC and VRTC continue to run, Video Enable status flag is reset, and the "Start Display" command must be given to re-enable the display.

Load Cursor Position:

	OPERATION	DESCRIPTION	DATA BUS MSB							LSB
Command	Write FF21H	Load Cursor	1	0	0	0	0	0	0	0
Parameters	Write FF20H	Char Number	(Char. Position in Row)							
	Write FF20H	Row Number	(Row Number)							

Action - The 8275 is conditioned to place the next two parameter bytes into the cursor position registers. Status flags not affected.

Enable Interrupt Command:

			DATA BUS							
	OPERATION	DESCRIPTION	MSB							LSB
Command	Write FF21H	Enable Interrupt	1	0	1	0	0	0	0	0
No Parameters										

Action - The interrupt enable status flag is set and interrupts are enabled.

Disable Interrupt Command:

			DATA BUS							
	OPERATION	DESCRIPTION	MSB							LSB
Command	Write FF21H	Disable Interrupt	1	1	0	0	0	0	0	0
No parameters										

Action - Interrupts are disabled and the interrupt enable status flag is reset.

Preset Counters Command:

			DATA BUS							
	OPERATION	DESCRIPTION	MSB							LSB
Command	Write FF21H	Preset Counters	1	1	1	0	0	0	0	0
No parameters										

Action - The internal timing counters are preset, corresponding to a screen display position at the top left corner. Two character clocks are required for this operation. The counters will remain in this state until any other command is given.

Status Flags

			DATA BUS							
	OPERATION	DESCRIPTION	MSB							LSB
Command	Read FF21H	Status Word	0	IE	IR	LP	IC	VE	DU	FO

IE - (Interrupt Enable) Set or reset by command. It enables vertical retrace interrupt. It is automatically set by a "Start Display"

command and reset wit the "Reset" command.

IR - (Interrupt Request) This flag is set at the beginning of display of the last row of the frame if the interrupt enable flag is set. It is reset after a status read operation.

LP - This flag is set when the light pen input (LPEN) is activated and the light pen registers have been loaded. This flag is automatically reset after a status read.

IC - (Improper Command) This flag is set when a command parameter string is too long or too short. The flag is automatically reset after a status read.

VE - (Video Enable) This flag indicates that video operation of the CRT is enabled. This flag is set on a "Start Display" command, and reset on a "Stop Display" or "Reset" command.

DU - (DMA Underrun) This flag is set whenever a data underrun occurs during DMA transfers. Upon detection Of DU, the DMA operation is stopped and the screen is blanked until after the vertical retrace interval. This flag is reset after a status read.

FO - (FIFO Overrun) This flag is set whenever the FIFO is overrun. It is reset on a status read.

Typical CRT-Controller init value in MicroMikko

1. RESET

Write TO

FF21H	0	0	0	0	0	0	0	0	= 00H
FF20H	0	1	0	0	1	1	1	1	= 4FH
FF20H	0	0	0	1	1	0	0	0	= 18 H
FF20H	1	1	0	1	1	1	1	0	= DEH
FF20H	0	1	1	0	0	1	1	1	= 67H

This selects following:

- Normal rows
- 80 char's/row
- Vertical (retrace ROW count = 1
- Vertical row/frame = 25
- Underline placement =14
- Number of lines/char's row = 15
- Line counter mode = 1
- Field attribute mode = 1
- Cursor is Nonblinkin reverse Video block
- Number of chr. counts per HRTC is 16

2. START DISPLAY

Write TO FF21H

0	0	1	0	0	0	1	1	= 23H

- Number of char. clokcs between DMA bursts = 0
- No. of DMA cycles per burst = 8

3. NO INTERRUPTS

Write to FF21H

1	1	0	0	0	0	0	0	= C0H

4. PRESET COUNTERS

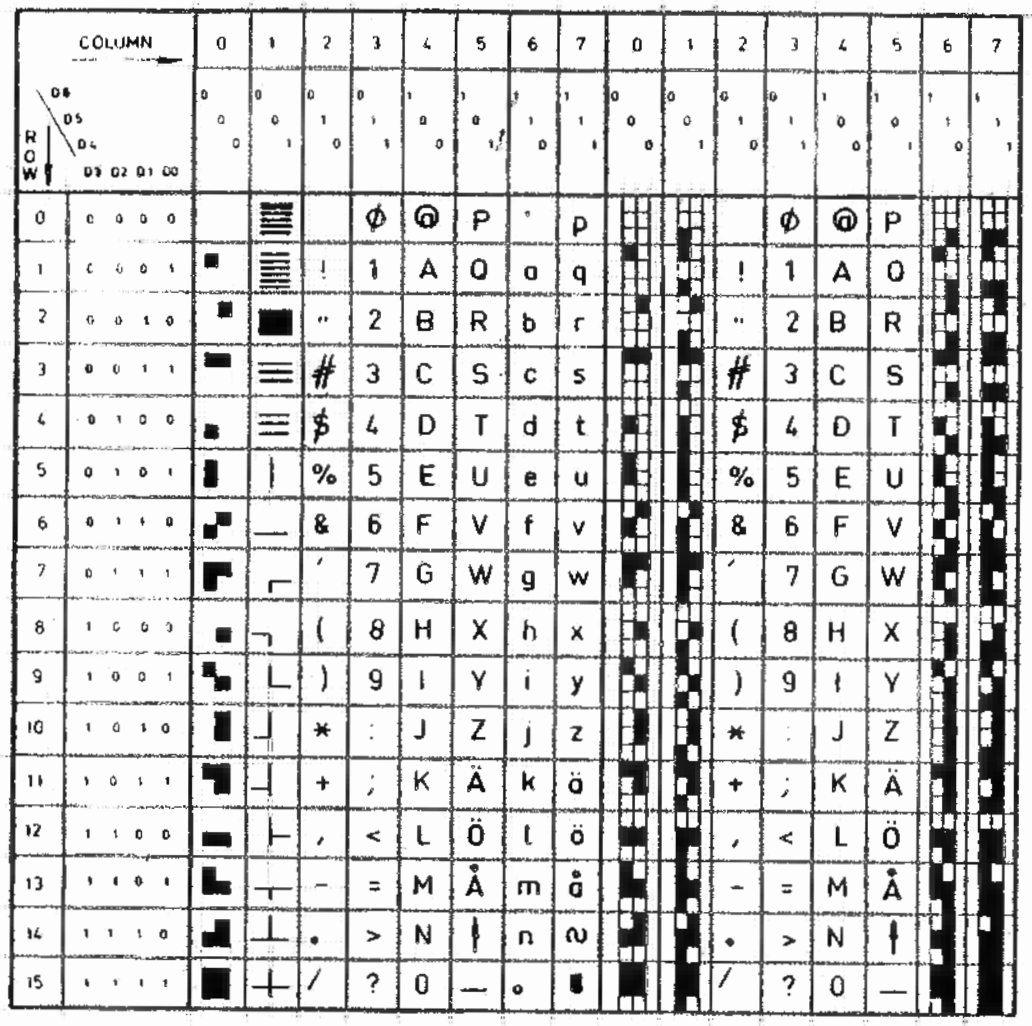
Write to FF21H

1	1	1	0	0	0	0	0	= E0H

5. LOAD CURSOR POSITION

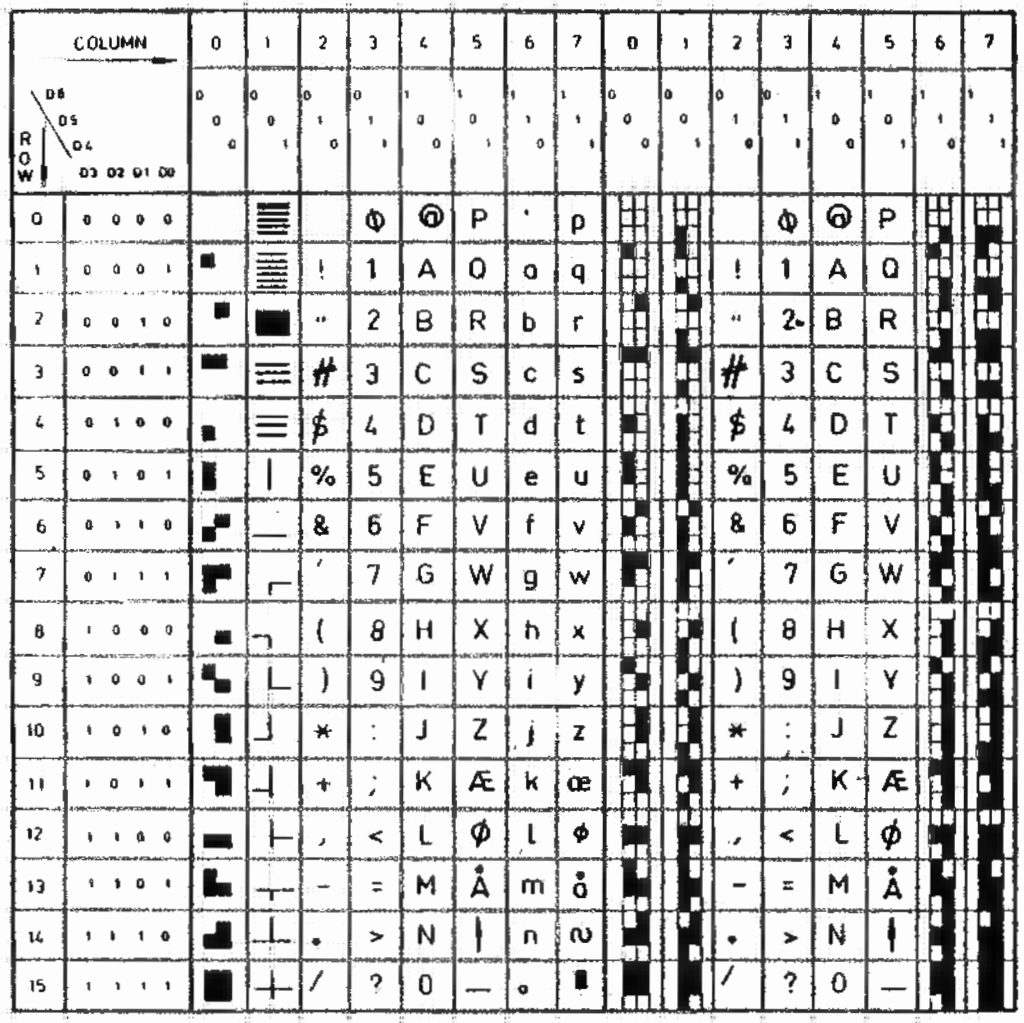
Write to

FF21H	1	0	0	0	0	0	0	0	= 80H
FF20H									= Char position in ROW (1-80)
FF20H									= Row number (1-24)



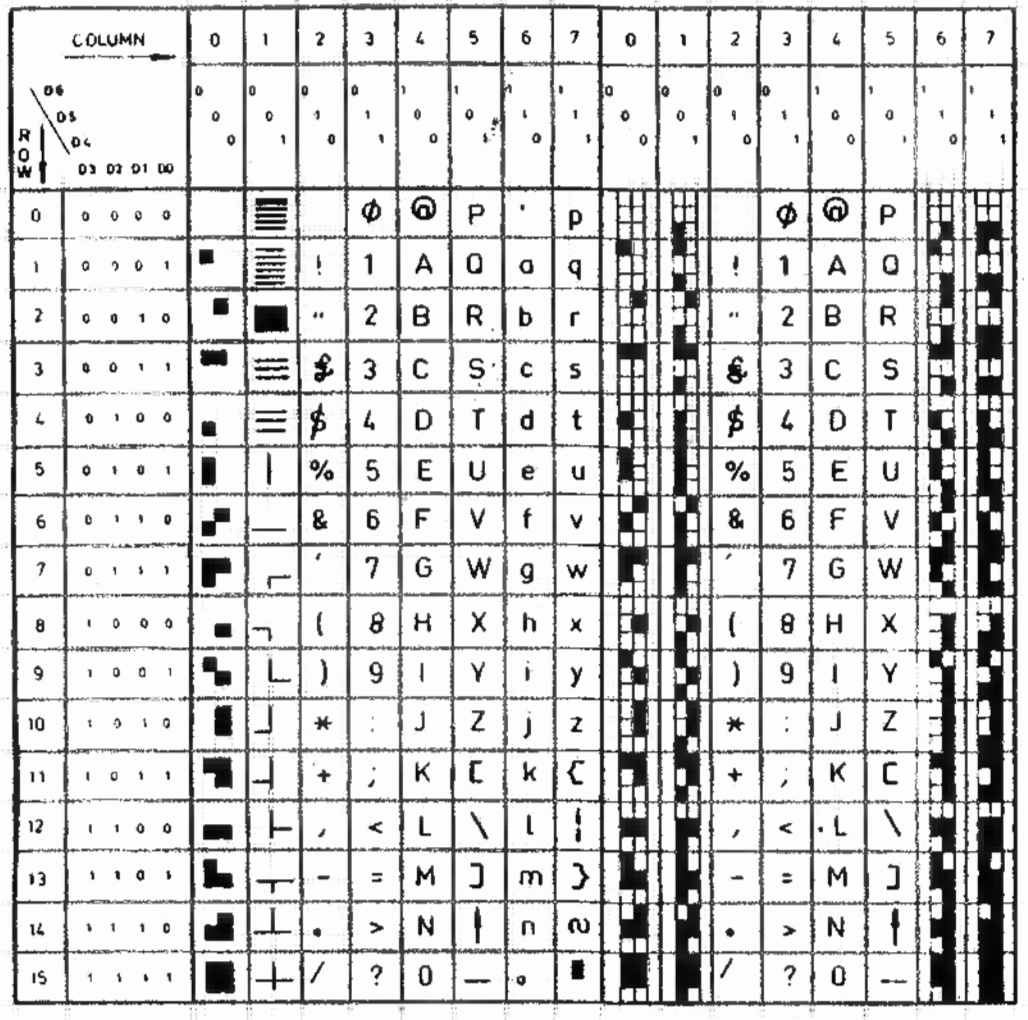
COLUMN → / ROW ↓ (D8 D5 D4 D3 D2 D1 D0)	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
0 (0 0 0 0)				ϕ	@	P	'	p				ϕ	@	P		
1 (0 0 0 1)	■		!	1	A	Q	a	q			!	1	A	Q		
2 (0 0 1 0)	■	■	"	2	B	R	b	r			"	2	B	R		
3 (0 0 1 1)	■	≡	#	3	C	S	c	s			#	3	C	S		
4 (0 1 0 0)	■	≡	$	4	D	T	d	t			$	4	D	T		
5 (0 1 0 1)	▌	▏	%	5	E	U	e	u			%	5	E	U		
6 (0 1 1 0)	▟	—	&	6	F	V	f	v			&	6	F	V		
7 (0 1 1 1)	▛	⌐	'	7	G	W	g	w			'	7	G	W		
8 (1 0 0 0)	■	¬	(	8	H	X	h	x			(	8	H	X		
9 (1 0 0 1)	▚	└	)	9	I	Y	i	y			)	9	I	Y		
10 (1 0 1 0)	▌	┘	*	:	J	Z	j	z			*	:	J	Z		
11 (1 0 1 1)	▛	┤	+	;	K	Ä	k	ä			+	;	K	Ä		
12 (1 1 0 0)	▬	├	,	<	L	Ö	l	ö			,	<	L	Ö		
13 (1 1 0 1)	▙	┼	-	=	M	Å	m	å			-	=	M	Å		
14 (1 1 1 0)	▟	┴	.	>	N	▌	n	ω			.	>	N	▌		
15 (1 1 1 1)	■	┼	/	?	O	—	∘	▮			/	?	O	—		

MIKROMIKKO 1 CHARACTER GENERATOR
SF/S 726807B

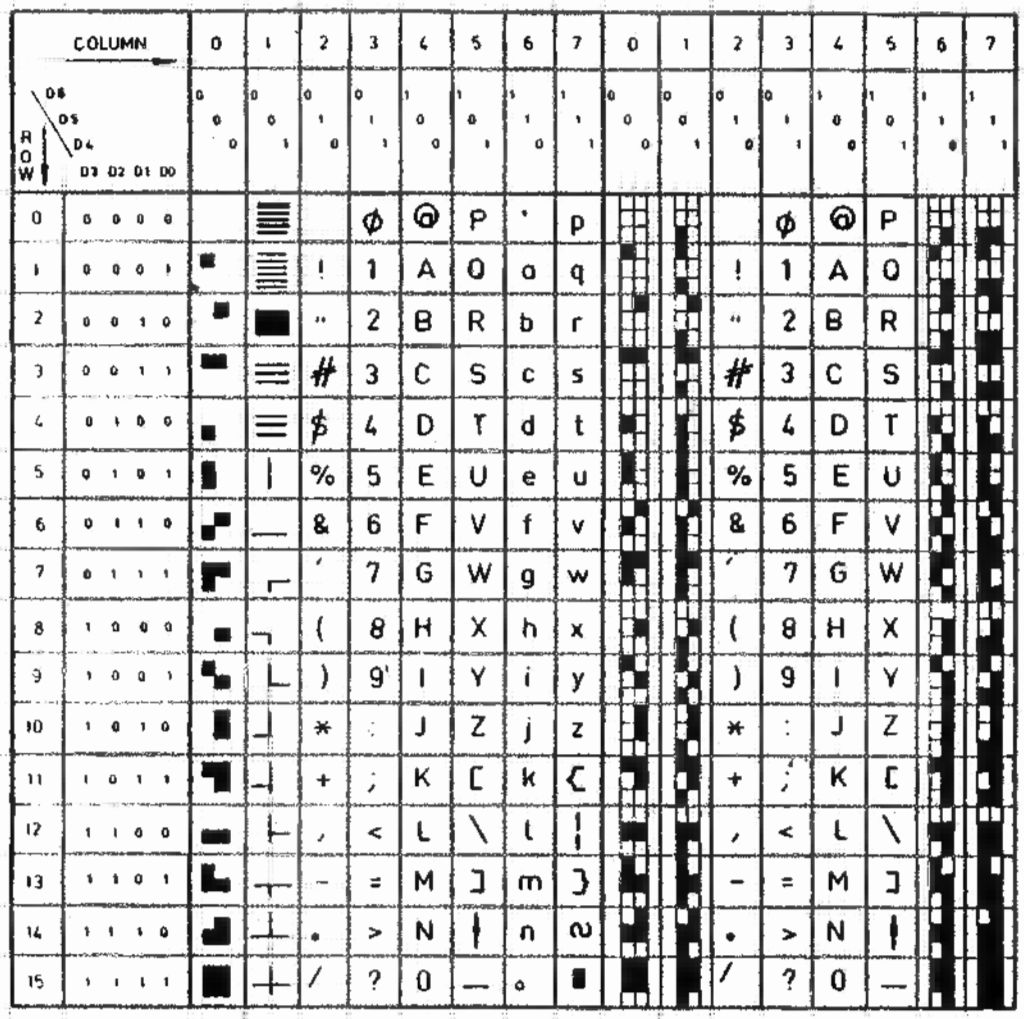


COLUMN → / ROW ↓ (D8 D5 D4 D3 D2 D1 D0)	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
0 (0 0 0 0)				Φ	@	P	'	p				Φ	@	P		
1 (0 0 0 1)	■		!	1	A	Q	a	q			!	1	A	Q		
2 (0 0 1 0)	■	■	"	2	B	R	b	r			"	2	B	R		
3 (0 0 1 1)	■	≡	#	3	C	S	c	s			#	3	C	S		
4 (0 1 0 0)	■	≡	¤	4	D	T	d	t			¤	4	D	T		
5 (0 1 0 1)	▌	▏	%	5	E	U	e	u			%	5	E	U		
6 (0 1 1 0)	▟	—	&	6	F	V	f	v			&	6	F	V		
7 (0 1 1 1)	▛	⌐	'	7	G	W	g	w			'	7	G	W		
8 (1 0 0 0)	■	¬	(	8	H	X	h	x			(	8	H	X		
9 (1 0 0 1)	▚	└	)	9	I	Y	i	y			)	9	I	Y		
10 (1 0 1 0)	▌	┘	*	:	J	Z	j	z			*	:	J	Z		
11 (1 0 1 1)	▛	┤	+	;	K	Æ	k	æ			+	;	K	Æ		
12 (1 1 0 0)	▬	├	,	<	L	Ø	l	ø			,	<	L	Ø		
13 (1 1 0 1)	▙	┼	-	=	M	Å	m	å			-	=	M	Å		
14 (1 1 1 0)	▟	┴	.	>	N	▌	n	ω			.	>	N	▌		
15 (1 1 1 1)	■	┼	/	?	O	—	∘	▮			/	?	O	—		

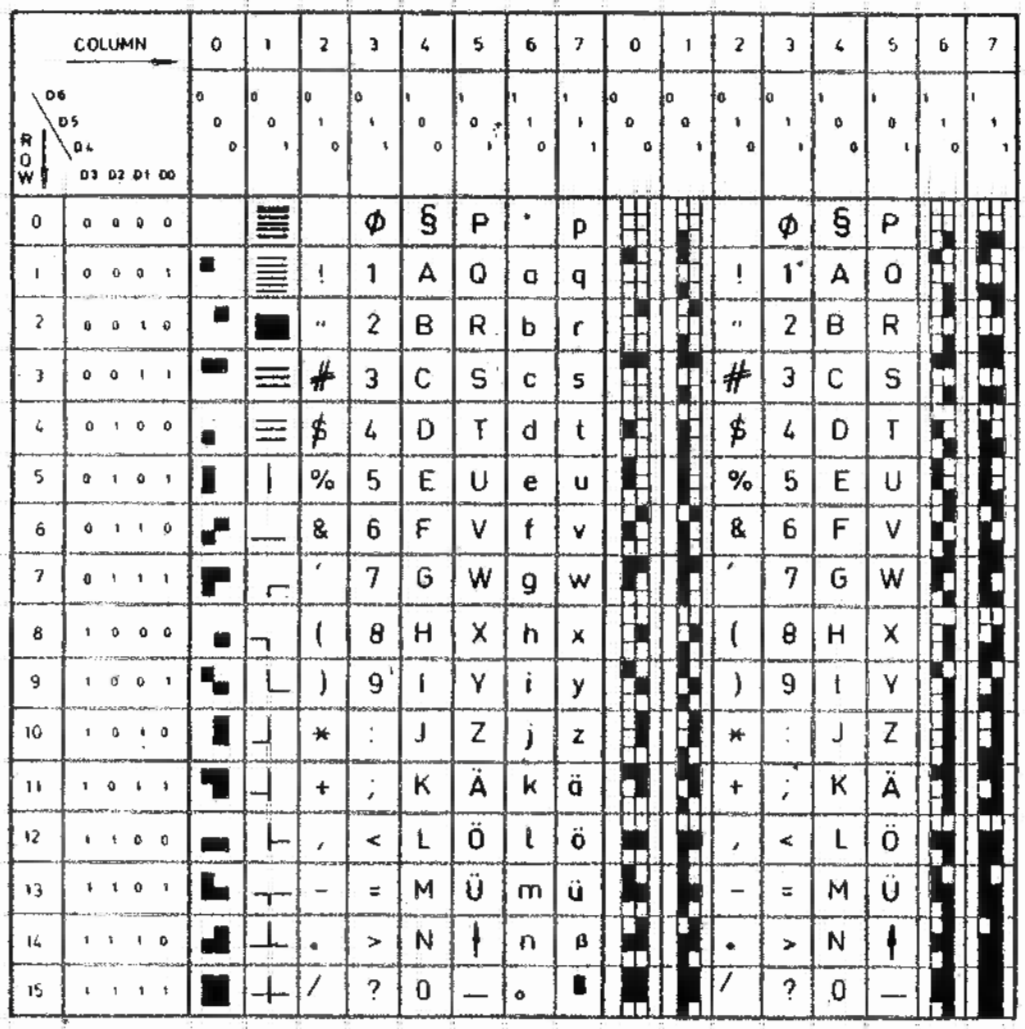
MIKROMIKKO 1 CHARACTER GENERATOR
N/DK 726806B



MIKROMIKKO 1 CHARACTER GENERATOR
GB 726803B



MIKROMIKKO 1 CHARACTER GENERATOR
US-ASCII 726876B



MIKROMIKKO 1 CHARACTER GENERATOR
D 726805B

2.5 BAUD RATE GENERATOR

Baud rate generator is used to support MPSC with clock freguencies. Location of baud rate generator is as follows.

FF30	ITXC	Main Channel, internal Transmit Clock
FF31	IRXC	Main Channel, internal Receive Clock
FF32	AUXC	Help Channel, Receive/Transmit Clock
FF33	Timer Conrol Register	

Line speed characteristics are set as follows.

Line speed/ Baud	Set value			
	sync		async	
	Hexad.	desim.	Hexad.	desim.
75	A000	40960	500	1280
110	5000	20480	369	873
150	2800	10240	280	640
300	1400	5120	140	320
600	A00	2560	A0	160
1200	500	1280	50	80
2400	280	640	28	40
4800	140	320	14	20
9600	A0	160	A	10

The IC of type INTEL 8253 is used. It supports three counters and has a Control Word register.

Counter description

Each Counter consists of a single, 16-bit, pre-settable, DOWN counter. The counter can operate in either binary or BCD and its input, gate and output are configured by the selection of MODES stored in the Control Word Register.

The counters are fully independent and each can have separate Mode configuration and counting operation, binary or BCD. Also, there are special features in the control word that handle the loading of the count value so that software overhead can be minimized for these functions.

The reading of the contents of each counter is available to the programmer with simple READ operations for event counting applications and special commands and logic are included in the 8253 so that the contents of each counter can be read "on the fly" without having to inhibit the clock input.

Control Word Register

The Control Word Register is selected with address FF33H. It then accepts information from the data bus buffer and stores it in a register. The information stored in this register controls the operational MODE of each counter, selection of binary or BCD counting and the loading of each count register.

The Control Word Register can only be written into; no read operation of its contents is available.

OPERATIONAL DESCRIPTION

General

The complete functional definition of the 8253 is programmed by the systems software. A set of control words must be sent out by the CPU to initialize each counter of the 8253 with the desired MODE and quantity information. These control words program the MODE, Loading sequence and selection of binary or BCD counting.

Once programmed, the 8253 is ready to perform whatever timing tasks it is assigned to accomplish.

Programming the 8253

All of the MODES for each counter are programmed by the systems software by simple I/O operations.

Each counter of the 8253 is individually programmed by writing a control word into the Control Word Register.

Control Word Format

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
SC1	SC0	RL0	RL1	M2	M1	M0	BCD

Definition of Control

SC - Select Counter:

SC1	SC0	
0	0	Select Counter 0
0	1	Select Counter 1
1	0	Select Counter 2
1	1	Illegal

RL - Read/Load:

RL1	RL0	
0	0	Counter Latching operation (see READ/WRITE Procedure Section)
1	0	Read/Load most significant byte only
0	1	Read/Load least significant byte only
1	1	Read/Load least isgnificant byte first, then most significant byte

M - MODE:

M2	M1	M0	
0	0	0	Mode 0
0	0	1	Mode 1
X	1	0	Mode 2
X	1	1	Mode 3
1	0	0	Mode 4
1	0	1	Mode 5

BCD:

0	Binary Counter 16-bits
1	Binary Coded Decimal (BCD) Counter (4 Decades)

Counter Loading

The count register is not loaded until the count value is written (one or two bytes, depending on the mode selected by the RL bits), followed by a rising edge and a falling edge of the clock. Any read of the counter prior to that falling clock edge may yeild invalid data.

MODE Definition

MODE 0 :**Interrupt onTerminal Count.** The output will be initially low after the mode set operation. After the count is loaded into the selected count register, the output will remain low and the counter will count. When terminal count is reached the output will go high and remain high until the selected count register is reloaded with the mode or a new count is loaded. The counter continues to decrement after terminal count had been reached.

Rewritting a counter register during counting results in the following

(1) Write 1st byte stops the current counting.
(2) Write 2nd byte starts the new count.

MODE 1: Programmable One-Shot. The output will go low on the count following the rising edge of the gate input.

The output will go high on the terminal count. If a new count value is loaded while the output is low it will not affect the duration of the one-shot pulse until the succeeding trigger. The current count can be read at any time without affecting the on-shot pulse.

The one-shot is retriggerable, hence the output will remain low for the full count after any rising edge of the gate input.

MODE 2: Rate Generator. Divide by N counter. The output will be low for one period of the input clock. The period from one output pulse to the next equals the number of input counts in the count register. If the count register is reloaded between output pulses the present period will not be affected, but the subsequent period will reflect the new value.

The gate input, when low, will force the output high. When the gate input goes high, the counter will start from the initial count. Thus, the gate input can be used to synchronize the counter.

When this mode is set, the output will remain high until after the count register is loaded. The output then can also be synchronized by software.

MODE 3: Square Wave Rate Generator. Similar to MODE 2 except that the output will remain high until one half the count has been completed (for even numbers) and go low for the other half of the count. This is accomplished by decrementing the counter by two on the falling edge of each clock pulse. When the counter reaches terminal count, the state of the output is changed and the counter is reloaded with the full count and the whole process is repeated.

If the count is odd and the output is high, the first clock pulse (after the count is loaded) decrement the count by 1. Subsequent clock pulses decrement the clock by 2. After timeout, the output goes low and the full count is reloaded. The first clock pulse (following the reload) decrement the counter by 3. Subsequent clock pulses decrement the count by 2 until timeout. Then the whole process is repeated. In this way, if the count is odd, the output will be high for (N + 1)/2 counts and low for (N - 1)/2 counts.

Mode 4: Software Triggered Strobe. After the mode is set, the output will be high. When the count is loaded, the counter will begin counting. On terminal count, the output will go low for one input clock period, then will go high again.

If the count register is reloaded between output pulses the present period will not be affected, but the subsecuent period will reflect the new value. The count will be inhibited while the gate input is low. Reloading the counter register will restart counting beginning with the new number.

MODE 5: Hardware Triggered Strobe. The counter will start counting after the rising edge of the trigger input and will go low for one clock period when the terminal count is reached. The counter is retriggerable. The output will not go low until the full count after the rising edge of any trigger.

Modes	Signal Status	Low Or Going Low	Rising	High
0		Disables counting	--	Enables counting
1		--	1) Initiates counting 2) Resets output after next clock	--
2		1) Disables counting 2) Sets output immedistely high	Initiates counting	Enables counting
3		1) Disables counting 2) Sets output immediately high	Initiates counting	Enables counting
4		Disables counting	--	Enables counting
5		--	Initiates counting	--

Gate Pin Operations Summary

With system initialization following parameters are set to obtain asynchronous 9600 bit/s rate.

Mode selection is 2. Read/load is set to two bytes 16-bit binary counter.

Address	R/W	operand
FF33H	W	36H
FF30H	W	OAH
FF30H	W	OOH
FF33H	W	40H+36H
FF31H	W	OAH
FF31H	W	OOH
FF33H	W	80H+36H
FF32H	W	OAH
FF32H	W	OOH

2.6 MPSC

All communication from/to outer world with MicroMikko is done via serial I/O-ports. Two pieces is provided, Main channel and AUX-channel. The Connector with AUX Channel is meant for asynchronous use, while Main channel provides full synchronous data transfer support. Main channel can use DMA, channels two and three, AUX channel is interrupt driven.

The whole serial I/O system consists of MPSC, DMA, and I/O line drivers. These I/O line drivers can be selected with 8-bit dataregister. DMA, if used, must be programmed to communicate with MPSC.

GENERAL DESCRIPTION

The Intel 8274 or NEC 7201 Multi-Protocol Serial Controller is a microcomputer peripheral device which supports asynchronous (Start/Stop), Byte Synchronous (Monosync, IBM Bisync), and Bit Synchronous (ISO's HDLC, IBM's SDLC) protocols. This controller's flexible architecture allows easy implementation of many variations of these three protocols with low software and hardware overhead.

The Multi-Protocol Serial Controller (MPSC) implements two independent serial receiver/transmitter channels.

The MPSC supports several microprocessor interface options; Polled, Wait, Interrupt driven and DMA driven. The MPSC is designed to support Intel's MCS-85 family.

FUNCTIONAL DESCRIPTION

This section describes how the Asynchronous and Synchronous protocols are implemented in the MPSC. It describes general considerations, transmit operation, and receive operation for Asynchronous, Byte Synchronous, and Bit Synchrounous protocols.

ASYNCHRONOUS OPERATIONS

General

For operation in the asynchronous mode, the MPSC must be initialized with the following information: character length (WR3; D7, D6 and WR5; D6 D5),clock rate (WR4; D7,D6), number of stop bits (WR4; D3, D2), odd, even or no parity (WR4; D1, D0), interrupt mode (WR1, WR2), and receiver (WR3; D0) or transmitter (WR5; D3) enable. When loading these prameters into the MPSC, WR4 information must be written before the WR1, WR3, WR5 parameters commands. (See Detailed Command Description Section).

For transmission via a modem or RS232C interface, the Request To Send (RTS) (WR5; D1) and Data Terminal Ready (DTR) (WR5; D7) bits must be set along with the Transmit Enable bit (WR5; D3). Setting the Auto Enables (WR3; D5) bit allows the programmer to send the first character of the message without waiting for a clear to send (CTS).

Both the Framing Error and Receive Overrun Error flags are latched and cause an interrupt, i.e., if status affects vector (WR1B; D2) is selected, the interrupt vector indicates a special Receive condition.

If the External/Status Interrupt bit (WR1; D0) is enabled, Break Detect (RR0; D7) and Carrier Detect (RR0; D3) will cause an interrupt. Reset External/Status Interrupts (WR0; D5, D4, D3) will clear Break Detect and Carrier Detect bits if they are set.

A status read after a data read will include error status for the next word in the buffer. If the Interrupt on First Character (WR1; D4, D3) is selected, then data and error status are held until an Error Reset command (WR0; D5, D4, D3) is given.

If the Interrupt on Every Character Mode bit (WR1; D4, D3) is selected, the interrupt vector is different if there is an error status in RR1. When the character is read, the error status bit is set and the Special Receive Condition vector is returned if Status Affects vector (WR1B; D2) is selected.

In a polled environment, the Receive Character Available bit (RR0; D0) must be monitored so that the CPU can determine when data is available. The bit is reset automatically when the data is read.

If the X1 clock mode is selected, the bit synchronization must be accomplished externally, and this is not possible in MicroMikko.

Transmit

The transmit function begins when the Transmit Enable bit (WR5; D3) is set. The MPSC automatically adds the start bit, the programmed parity bit (odd, even or no parity) and the programmed number of stop bits(1, 1.5 or 2 bits) to the data character being transmitted.

The Serial data is shifted out from the Transmit Data (TxD) output on the falling edge of the Transmit Clock (TxC) input, at a rate programmable to 1, 1/16, 1/32nd, or 1/64th of the clock rate supplied from the Baud rate generator.

The TxD output is held high when the transmitter has no data to send, unless, under program control, the Send Break (WR5; D4) command is issued to hold the TxD low.

If the External/STATUS Interrupt bit (WR1; D0) is set the status of Carrier Detect (CD) , Clear To Send (CTS) and Synchronous Detection (SYNDET) are monitored, and, if any changes occur for a period of time greater than the minimum specified pulse width, an interrupt is generated. CTS is usually monitored using this interrupt feature.

If the Auto Enables (WR; D5) option is selected the programmer need not wait for the CTS before sending the first character. The MPSC will automatically wait for the CTS pin to go active before the transmission begins.

The Transmit Buffer Empty bit (RR0; D2) is set by the MPSC when the data byte from the buffer is loaded in the transmit shift register. The data is written to the MPSC only when the Tx buffer becomes empty to prevent overwriting.

Receive

The receive function begins when the receive Enable (WR3; D0) bit is set. If the Auto Enabled (WR3; D5) option is selected, then Carrier Detect (CD) must also be low. A valid start bit is detected if a low persists for at least 1/2 bit time on the Receive Data (RxD) input.

The data is sampled at mid-bit time, on the rising edge of RxC, until the entire character is assembled. The receiver inserts 1's when a character is less than 8 bits. If parity (WR4; D1, D0) is enabled and the character is less than 8 bits the parity bit is not stripped from the character.

The receiver also stores error status for each of the 3 data characters in the data buffer. When a parity error is detected, the parity error flag (RR1; D4) is set and remains set until it is reset by the Error Reset command (WR0; D5, D4, D3).

When a character is assembled without a stop bit being detected, the Framing Error bit (RR1; D6) is set. The detection of a Framing Error adds an additional 1/2 bit time to the character time so the Framing Error is not interpreted as a new start bit.

If the CPU fails to read a data character while more than three characters have been received, the Receive Overrun bit (RR1; D5) is set. Only the overwritten character is flagged with the Receive Overrun bit. When this occurs, the foutrh character assembled replaces the third character in the receive buffers. The Receive Overrun bit (RR1; D5) is reset by the Error Reset command (WR0; D5, D4, D3).

Asynchronous Mode Register Setup

	D7	D6	D5	D4	D3	D2	D1	D0
WR3	00-5 01-7 10-6 11-8	bit per received character	AUTO ENABLES	O	O	O	O	RX ENABLES
WR4	00-*1 01-*16 10-*32 11-*64	clock	O	O	00-ENABLE SYNC MODE 01-1 stop bit 10-1,5 stop bits 11-2 stop bits		PARITY 0=ODD 1=EVEN	PARITY ENABLE
WR5	DTR	00-5 bits per 01-7 transmitted 10-6 11-8 charactor		SEND BREAK	Tx Enable	O	RTS	O

SYNCHRONOUS OPERATION - MONO SYNC, BI SYNC

General

The MPSC must be initialized with the following parameters: odd or even parity (WR4; D1, D0), X1 clock mode (WR4; D7, D6), 8- or 16-bit sync character (WR4; D5, D4), CRC polynomial (WR5; D2), Transmitter Enable (WR5; D3), interrupt modes (WR1, WR2), transmit character length (WR5; D6, D5) and receive character length (WR3; D7, D6). WR4 parameters must be written before WR1, WR3, WR5, WR6 and WR7.

The data is transmitted on the falling edge of the Transmit Clock, (TxC) and is received on the rising edge of Receive Clock (RxC). The X1 clock is used for both transmit and receive operations for all three sync modes: Mono, Bi and External.

Transmit Set-Up - Monosync, Bisync

Transmit data is held high after channel reset, or if the transmitter is not enabled. A break may be programmed to generate a spacing line that begins as soon as the Send Break (WR5; D4) bit is set. With the transmitter fully initialized and enabled, the default condition is continouos transmission of the 8- or 16-bit sync character.

Using interrupts for data transfer requires that the Transmit Interrupt/DMA Enable bit (WR1; D1) be set. An interrupt is generated each time the transmit buffer becomes empty. The interrupt can be satisfied either by writing another character into the transmitter or by resetting the Transmitter Interrupt/DMA Pending latch with a Reset Transmitter Interrupt/DMA Pending Command (WR0; D5, D4, D3). If nothing more is written into the transmitter, there can be no further Transmit Buffer Empty interrrupt, but this situation does cause a Transmit Underrun condition (RR0; D6).

DMA data transfers use the TxDRQ A/B signals which indicate that the transmit buffer is empty, and that the MPSC is ready to accept the next data character. If the data character is not loaded into the MPSC by the time the transmit shift register is empty, the MPSC enters the Transmit Underrun condition.

The MPSC has two programmable options for solving the transmit underrun condition: it can insert sync characters, or it can send the CRC characters generated so far, followed by sync characters. Following a chip or channel reset, the Transmit Underrun/EOM status bit (RR0; D6) is in a set condition allowing the insertion of sync characters when there is no data to send. The CRC is not calculated on these automatically inserted sync characters. When the CPU detects the end of message, a Reset Transmit Underrun/EOM command can be issued. This allows CRC to be sent when the transmitter has no data to send.

In the case of sync insertion, an interrupt is generated only after the first automatically inserted sync character has been loaded in Transmit Shift Register. The status indicates the Transmit Underrun/EOM bit and the Transmit Buffer Empty bit are set.

In the case of CRC insertion, the Transmit Underrun/EOM bit is set and the Transmit Buffer Empty bit is reset while CRC is being sent. When CRC has been completely sent, the Transmit Buffer Empty status bit is set and an interrupt is generated to indicate to the CPU that another message can begin (this interrupt occurs because CRC has been sent and sync has been loaded into the Tx Shift Register). If no more messages are to be sent, the program can terminate transmission by resetting RTS, and disabling the transmitter (WR5; D3).

Bisync CRC Generation, Setting the Transmit CRC enable bit (WR5; D0) indicates CRC accumulation when the program sends the first data character to the MPSC. Although the MPSC automatically transmits up to two sync characters (16 bit sync), it is wise to send a few more sync characters ahead of the message (before enabling Transmit CRC) to ensure synchronization at the receiving end.

The Transmit CRC Enable bit can be changed on the fly any time in the message to include or exclude a particular data character from CRC accumulation. The Transmit CRC Enable bit should be in the desired state when the data character is loaded from the transmit shift register. To ensure this bit in the proper state, the Transmit CRC Enable bit must be issued before sending the data character to the MPSC.

Synchronous Mode Register Setup—Monosync, Bisync

	D7	D6	D5	D4	D3	D2	D1	D0
WR3	00 Rx 5 b/char 01 Rx 7 b/char 10 Rx 6 b/char 11 Rx 8 b/char		AUTO ENABLES	ENTER HUNT MODE	Rx CRC ENABLE	0	SYNC CHAR LOAD INHIBIT	Rx ENABLE
WR4	0	0	00 8 bit Sync 01 16 bit Sync 11 Ext Sync		0	0	EVEN/$\overline{\text{ODD}}$ PARITY	PARITY ENABLE
WR5	DTR	00 Tx 5 b/char 01 Tx 7 b/char 10 Tx 6 b/char 11 Tx 8 b/char		SEND BREAK	Tx ENABLE	1 (SELECTS CRC-16)	RTS	Tx CRC ENABLE

Transmit Transparent Mode. Transparent mode (Bisync protocol) operation is made possible by the ability to change Transmit CRC Enable on the fly and by the additional capability of inserting 16 bit sync characters. Exclusion of DLE characters from CRC calculation can be achieved by disabling CRC calculation immediately preceding the DLE character transfer to the MPSC.

In the transmit mode, the transmitter always sends the programmed number of sync bits (8 or 16) (WR4; D5, D4). When in the Monosync mode, the transmitter sends from WR6 and the receiver compares against WR7. One of two CRC polynomials, CRC 16 or SDLC, may be used with synchronous modes. In the transmit initialization process, the CRC generator is initialized by setting the Reset Transmit CRC Generator command (WR0; D7, D6).

The External/Status interrupt (WR1; D0) mode can be used to monitor the status of the CTS input as well as the Transmit Underrun/EOM latch. Optionally, the Auto Enable (WR3; D5) feature can be used to enable the transmitter when CTS is active. The first data transfer to the MPSC can begin when the External/Status interrupt occurs (CTS (RR0; D5) status bit set) following the Transmit Enable command (WR5; D3).

Receive

After a channel reset, the receiver is in the Hunt phase, during which the MPSC looks for character synchronization. The Hunt begins only when the receiver is enabled and data transfer begins only when character synchronization has been achieved. If character synchronization is lost, the hunt phase can be re-entered by writing the Enter Hunt Phase (WR3; D4) bit. The assembly of received data continue until the MPSC is reset or until the receiver is disabled (by command or by CD while in the Auto Enables mode) or until the CPU sets the Enter Hunt Phase bit. Under program control, all the leading sync characters of the message can be inhibited from loading the receive buffers by setting the Sync Character Load Inhibit (WR3; D1) bit. After character synchronization is achieved the assembled characters are transferred to the receive data FIFO.

Data may be transferred with or without interrupts. Transferring data without interrupts is used for a purely polled operation or for off-line conditions. There are three interrupt modes available for data transfer: Interrupt on First Character Only, Interrupt on Every Character, and Special Receiver Conditions Interrupt.

Interrupt on First Character Only mode is normally used to start a polling loop, or a DMA transfer using the RxDRQ signal. The MPSC interrupts on the first character and thereafter only interrupts after a Special Receive Condition is detected. This mode can be reinitialized using the Enable Interrupt On Next Receive Character (WR0; D5, D$, D3) command which allows the next character received to generate an interrupt. Parity Errors do not cause interrupts, but End of Frame (SDLC operation) and Receive Overrun do cause interrupts in this mode. If the external status interrupts (WR1; D0) are enabled an interrupt may be generated any time the CD changes state.

Interrupt On Every Character mode generates an interrupt whenever a character enters the receive buffer. Errors and Special Receive Conditions generate a special vector if the Status Affects Vector (WR1B; D2) is selected. Also the Parity Error may be programmed (WR1; D4, D3) not to generate the special vector while in the Interrupt on Every Character mode.

The Special Receive Condition interrupt can only occur while in the Receive Interrupt On First Character Only or the Interrupt on Every Receive Character modes. The Special Receive Condition interrupt is caused by the Receive Overrun

(RR1; D5) error condition. The error status reflects an error in the current word in the receive buffer, in addition to any Parity or Overrun errors since the last Error Reset (WR0; D5, D4, D3). The Receive Overrun and Parity error status bits are latched and can only be reset by the Error Reset (WR0; D5, D4, D3) command.

SYNCHRONOUS OPERATION - SDLC

General

Like the other synchronous operations the SDLC mode must be initialized with the following parameters: SDLC mode (WR4; D5, D4), SDLC polynomial (WR5; D2), Request to Send, Data Terminal Ready, transmit character length (WR5; D6, D5), interrupt modes (WR1; WR2), Transmit Enable (WR5; D3), Receive Enable (WR3; D0), Auto Enable (WR3; D5) and External/Status Interrupt (WR1; D0). WR4 parameters must be written before WR1, WR3, WR5, WR6 and WR7.

The Interrupt modes for SDLC operation are similar to those discussed preciously in the synchronous operations section.

Transmit

After a channel reset, the MPSC begins sending SDLC flags.

Following the flags in an SDLC operation the 8-bit address field, control field and information field may be sent to the MPSC by the microprocessor. The MPSC transmits the Frame Check Sequence using the Transmit Underrun feature. The MPSC automatically inserts a zero after every sequence of 5 consecutive 1's except when transmitting Flags or Aborts.

SDLC - like protocols do not have provision for fill characters within a message. The MPSC therefore automatically terminates an SDLC frame when the transmit data buffer and output shift register have no more bits to send. It does this by sending the two bytes of CRC and then one or more flags. This allows very high-speed transmissions under DMA or CPU control without requiring the CPU to respond quickly to the end-of message situation.

After a reset, the Transmit Underrun/EOM status bit is in the set state and prevents the insertion of CRC characters during the time there is no data to send. Flag characters are sent. The MPSC begins to send the frame when data is written into the transmit buffer. Between the time the first data byte is written, and the end of the message, the Reset Transmit Underrun/EOM (WR0; D5, D4, D1) command must be issued. The Transmit Underrun/EOM status bit (RR0; D6) is in the reset state at the end of the message which automatically sends the CRC characters.

The MPSC may be programmed to issue a send Abort command (WR0; D5, D4, D3). This command causes at least eight 1's but less than fourteen 1's to be sent before the line reverts to continuous flags.

Receive

After initialization, the MPSC enters the Hunt phase, and remains in the Hunt phase until the first Flag is received. The MPSC never again enters the Hunt phase unless the microprocessor writes the Enter Hunt command.

Synchronous Mode Register Setup—SDLC/HDLC

	D7	D6	D5	D4	D3	D2	D1	D0
WR3	00 Rx 5b/char 01 Rx 7b/char 10 Rx 6b/char 11 Rx 8b/char		AUTO ENABLES	ENTER HUNT MODE	Rx CRC ENABLE	ADDRESS SEARCH MODE	0	Rx ENABLE
WR4	0	0	1 (SELECTS SDLC/ HDLC MODE)	0	0	0	0	0
WR5	DTR	00 Tx 5b/char 01 Tx 7b/char 10 Tx 6b/char 11 Tx 8b/char		0	Tx ENABLE	0 (SELECTS SDLC/ HDLC CRC)	RTS	Tx CRC ENABLE

The MPSC can be programmed to receive all frames or it can be programmed to the Address Search Mode. In the Address Search Mode, only frames with addresses that match the value in WR6 or the global address (0FFH) are received by the MPSC. Extended address recognition must be done by the microprocessor software.

The control and information fields are received as data.

SDLC/HDLC CRC calculation does not have an 8-bit delay, since all characters are included in the calculation, unlike Byte Synchronous Protocols.

Reception of an abort sequence (7 or more 1's) will cause the Break/Abort bit (RR0; D7) to be set and will cause an External/Status interrupt, if enabled. After the Reset External/Status Interrupts Command has been issued, a second interrupt will occur at the end of the abort sequence.

MPSC

DETAILED COMMAND DESCRIPTION

General

The MPSC supports an extremely wide set of serial and system interface modes.

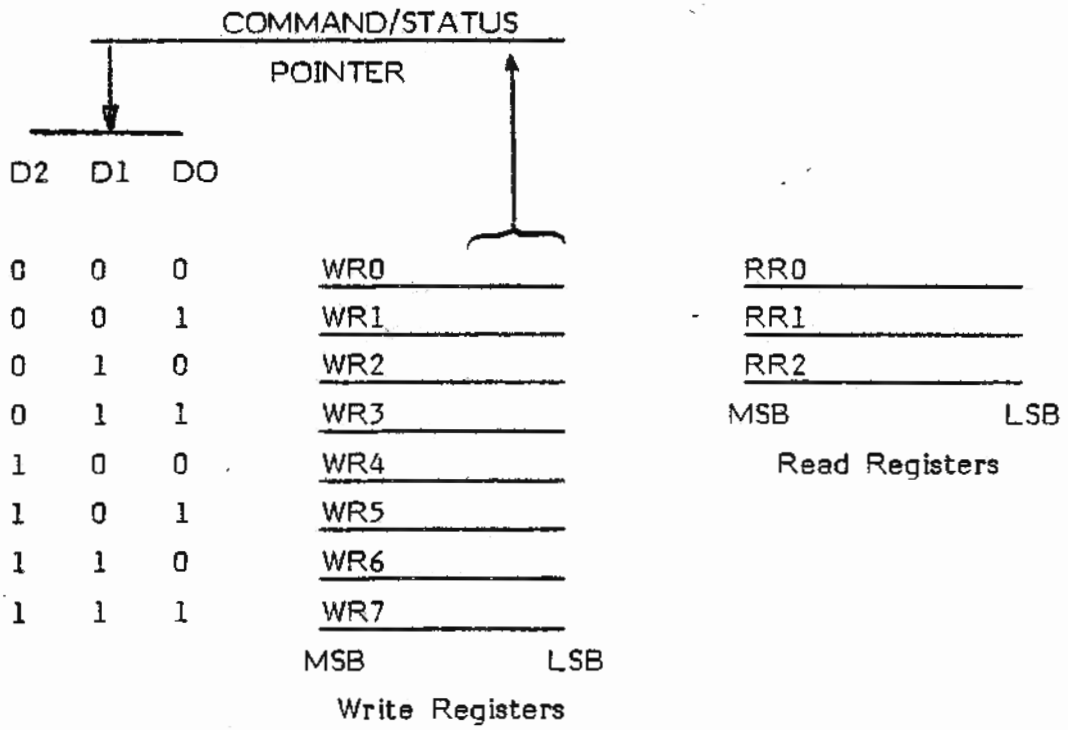
The system interface to the CPU consists of 8 ports of buffers:

ADRESS	Read Operation	Write Operation
FF10	ch. A Data Read	ch. A Data WRite
FF12	ch. A Status Read	ch. A Command/Parameter
FF11	ch. B Data Read	ch. B Data Write
FF13	ch. B Status Read	ch. B Command/Parameter

Data buffers are addressed by FF10H and FF11H and command ports are addressed by FF12H and FF13H.

Command, parameter, and status information is held in 22 registers within the MPSC (8 write registers and 3 read registers for each channel). They are all a accessed via the command ports.

An internal pointer register select which of the command or status registers will be read or written during a command/status access of an MPSC channel.



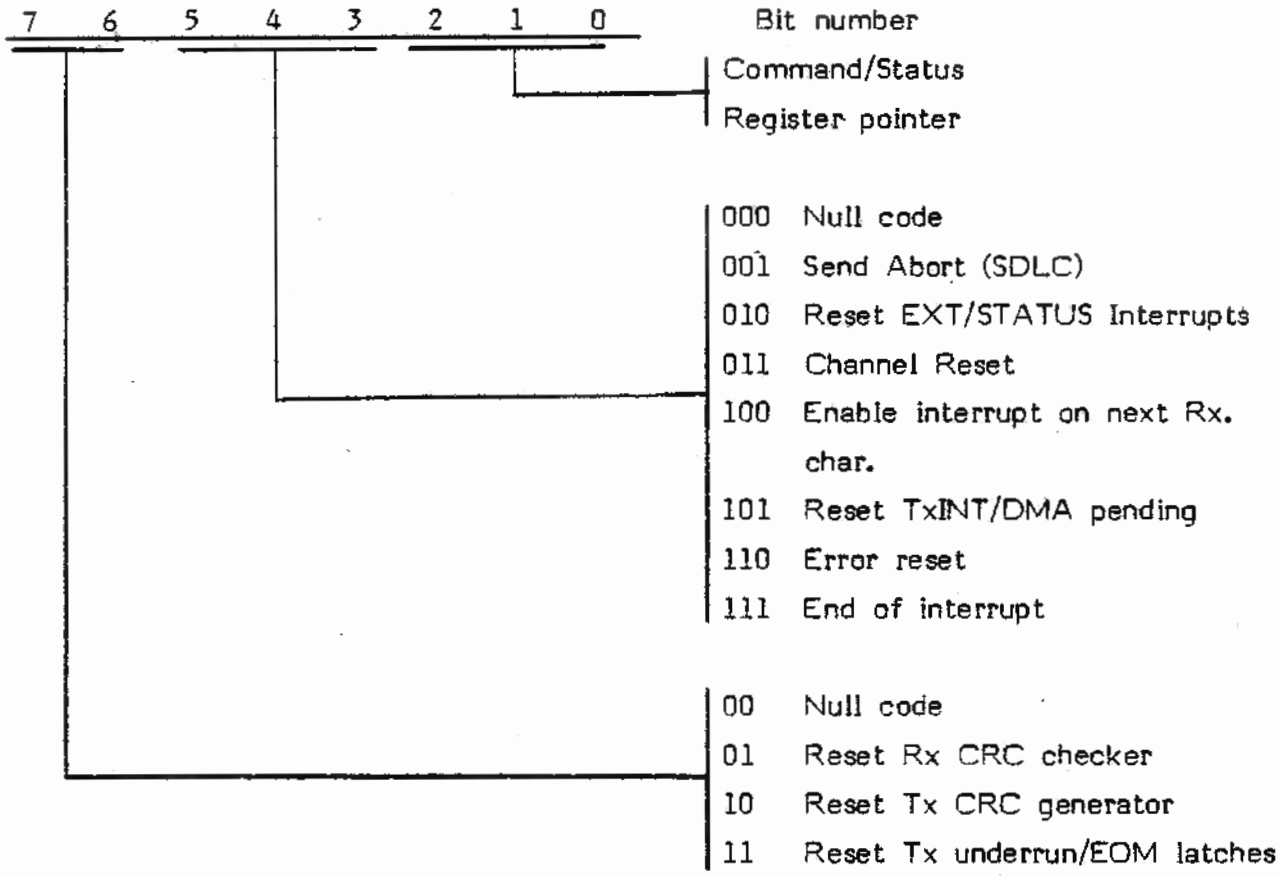
After reset, the contents of the pointer register are zero. The first write to a command register causes the data to be loaded into Write Register 0 (WR0). The three least significant bits of WR0 are loaded into the Command/Status Pointer. The next read or write operation accesses the read or write register selected by the pointer. The pointer is reset after the read or write operation is completed.

Command/Status Description

The following command and status bytes are used during initialization and execution phases of operation. All Command/Status operations on the two channels are identical, and independent, except where noted.

DETAILED REGISTER DESCRIPTION

WRITE REGISTER 0 (WR0)



D2, D1, D0 - Command/Status Register Pointer bits determine which write-register the next byte is to be written into, or which read-register the next byte is to be read from. After reset, the first byte written into either channel goes into WR0. Following a read or write to any register (except WR0) the pointer will point to WR0.

D5, D4, D3 - Command bits determine which of the basic seven commands are to be performed.

Command 0 Null - has no effect.

Command 1 Send Abort - causes the generation of eight to thirteen 1's when in the SDLC mode.

Command 2 Reset External/Status Interrupts - resets the latched status bits of RR0 and re-enables them, allowing interrupts to occur again.

Command 3 Channel Reset - resets the Latched Status bits of RR0, the interrupt prioritization logic and all control registers for the channel. Four extra system clock cycles should be allowed for MPSC reset time before any additional commands or controls are written into the channel.

Command 4 Enable Interrupt on Next Receive Character - if the Interrupt on First Receive Character mode is selected, this command reactivates that mode after each complete message is received to prepare the MPSC for the next message.

Command 5 Reset Transmitter Interrupt/DMA Pending - if The Transmit Interrupt/DMA Enable mode is selected, the MPSC automatically interrupts or requests DMA data transfer when the transmit buffer becomes empty. When there are no more characters to be sent, issuing this command prevents further transmitter interrupts or DMA requests until the next character has been completely sent.

Command 6 Error Reset - error latches, Parity and Overrun errors in RR1 are reset.

Command 7 End of Interrupt - resets the interrupt-in-service latch of the highest-priority internal device under service.

D7,D6 CRC Reset Code

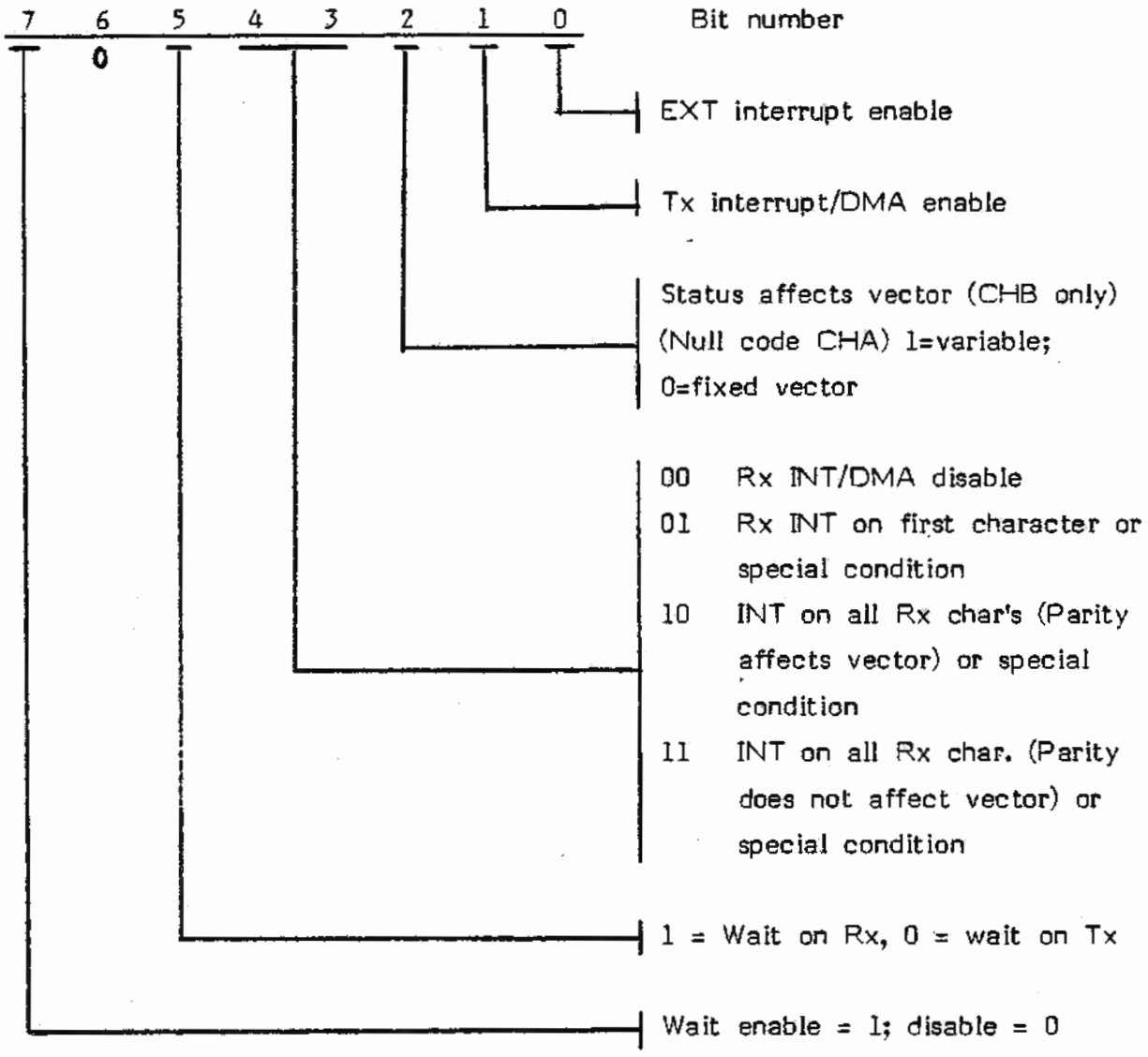
00 Null - has no effect

01 Reset Receive CRC Checker - resets the CRC checker to 0's. If in SDLC mode the CRC checker is initialized to all 1's.

10 Reset Transmit CRC Generator - resets the CRC generatior to 0's. If in SDLC mode the CRC generator's initialized to all 1'a.

11 Reset Tx Underrun/End of Message Latch

WRITE REGISTER 1 (WR 1)



D0 External/Status Interrupt Enable - allows interrupt to occur as the result of transitions on the CD, CTS or SYNDET inputs. Also allows interrupts as the result of a Break/Abort detection and termination, or at the beginning of CRC, or sync character transmission when the Transmit Underrun/EOM latch becomes set.

D1 Transmitter Interrupt/DMA Enable - allows the MPSC to interrupt or request a DMA transfer when the transmitter buffer becomes empty.

D2 — Status Affects vector - (WR1, D2 active in channel B only). If this bit is not set, then the fixed vector, programmed in WR2, is returned from an interrupt acknowledge sequence. If the bit is set then the vector returned from an interrupt acknowledge is variable as shown in the Interrupt Vector Table.

D4,D3 — Receive Interrupt Mode

0 0 — Receive Interrupt/DMA Disabled

0 1 — Receive Interrupt in First Character Only or Special Condition

1 0 — Interrupt on All Receive Characters or Special Condition (Parity Error is a Special Receive Condition)

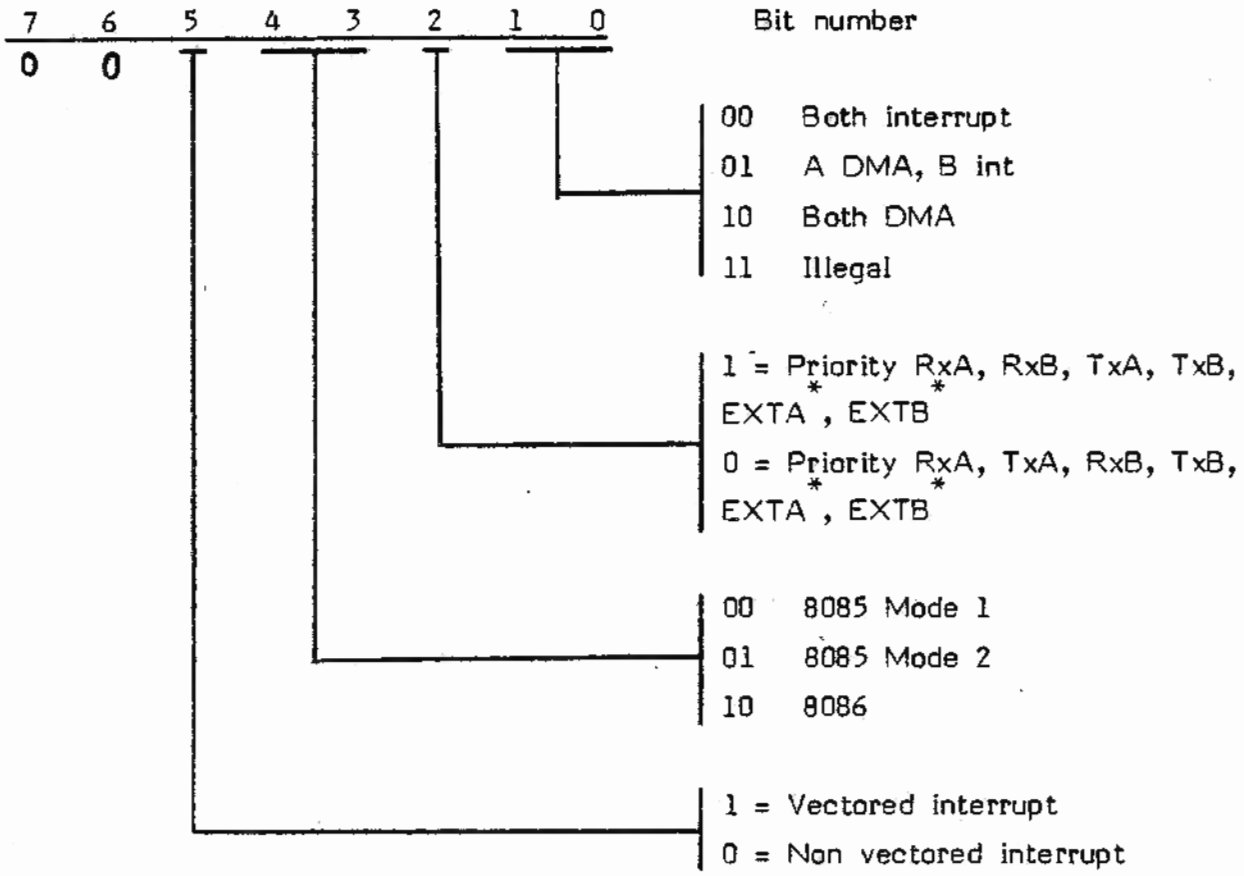
1 1 — Interrupt on All Receive Characters or Special Condition (Parity Error is not a Special Receive Condition).

D5 — Wait on Receive/Transmit - when the following conditions are met the RDY pin is activated, otherwise it is held in the High-Z state. (Conditions: Interrupt Enabled Mode, Wait Enabled, CS = 0, A0 = 0/1, and A1 = 0). The RDY pin is pulled low when the transmitter buffer is full or the receiver buffer is empty and it is driven High when the transmitter buffer is empty or the receiver buffer is full. The RDY_A and RDY_B may be wired OR connected since only one single is active at any one time while the other is in the High Z state.

D6 — Must be Zero

D7 — Wait Enable - enables the wait function.

WRITE REGISTER 2 (WR 2) Channel A



* External status interrupt.
Only if EXT interrrupt enable (WR1:D0) is set

D1,D0 System Configuration - These specify the data transfer from MPSC channels to the CPU, either interrupt or DMA based.

0 0 Channel A and Channel B both use interrupts

0 1 Channel A uses DMA, Channel B uses interrupt

1 0 Channel A and Channel B both use DMA

1 1 Illegal Code

D2 Priority - this bit specifies the relative priorities of the internal MPSC interrupt/DMA sources.

0 (Highest) RxA, TxA, RxB, ExTA, ExTB (Lowest)

1 (Highest) RxA, RxB, TxA, TxB, ExTa, ExTB (Lowest)

D5,D4,D3 Interrupt Code - specifies the behavior of the MPSC when it receives an interrupt acknowledge sequence from the CPU.

0 X X Non-vectored interrupts - intended for use with external DMA CONTROLLER. The Data Bus remains in a high impedence state during INTA sequences.

1 0 0 8085 Vector Mode 1 - intended for use as the primary MPSC in a daisy chained priority structure.

1 0 1 8085 Vector Mode 2 - intended for use as any secondary MPSC in a daisy chained priority structure.

1 1 0 8086/88 Vector Mode

D7,D6 Must be zero

WRITE REGISTER 2 (WR2): Channel B

MSB							LSB
V7	V6	V5	V4	V3	V2	V1	V0

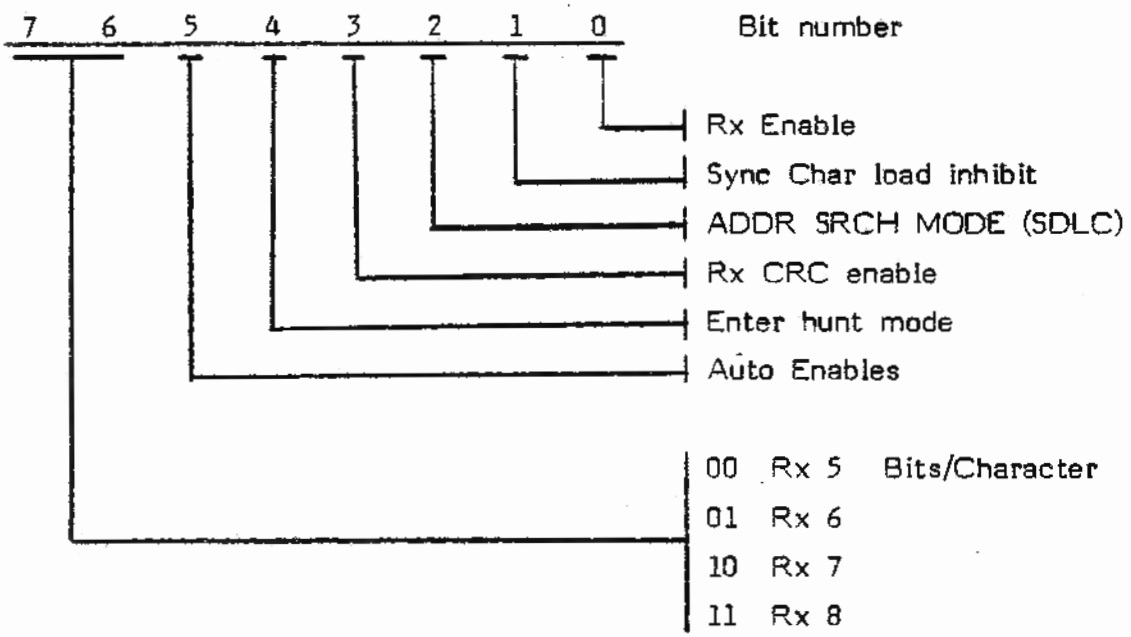
Interrupt
Vector

D7-D0 Interrupt vector - This register contains the value of the interrupt vector placed on the data bus during interrupt acknowledge sequence.

V4	V3	V2	Channel	Condition
0	0	0		Tx Buffer Empty
0	0	1	B	Ext/Status Change
0	1	0		Rx/Char available
0	1	1		Special Rx condition *
1	0	0		Tx Buffer empty
1	0	1	A	Ext/Status change
1	1	0		Rx Char. Available
1	1	1		Special Rx condition *

*Special receive condition
Parity error, Rx overrun error, Framing error, End of Frame (SDLC)

WRITE REGISTER 3 (WR3):



D0 Receiver Enable - A one enable the receiver to begin. This bit should be set only after the receiver has been initialized.

D1 Sync Character Load Inhibit - A one prevents the receiver from loading sync characters into the receive buffers.

D2 Address Search Mode - If the SDLC mode has been selected, the MPSC will receive all frames unless this bit is a 1. If this bit is a 1, the MPSC will receive only frames with address bytes that match the global address (0FFH) or the value loaded into WR6. This bit must be zero in non-SDLC modes.

Receive CRC Enable - A one in this bit enables (or re-enables) CRC calculation. CRC calculation starts with the last character placed in the Receiver FIFO. A zero in this bit disables, but does not reset, the Receiver CRC generator.

D4 Enter Hunt Phase - After initialization, the MPSC automatically enters the Hunt mode. If synchronization is lost, the Hunt phase can be re-entered by writing a one to this bit.

D5 Auto Enables - A one written to this bit causes CD to be automatic enable signal for the receiver and CTS to be an automatic enable signal for the transmitter. A zero written to this bit limits the effect of CD and CTS signals to setting/resetting their corresponding bits in the status register (RR0).

D7,D6 Receive Character length

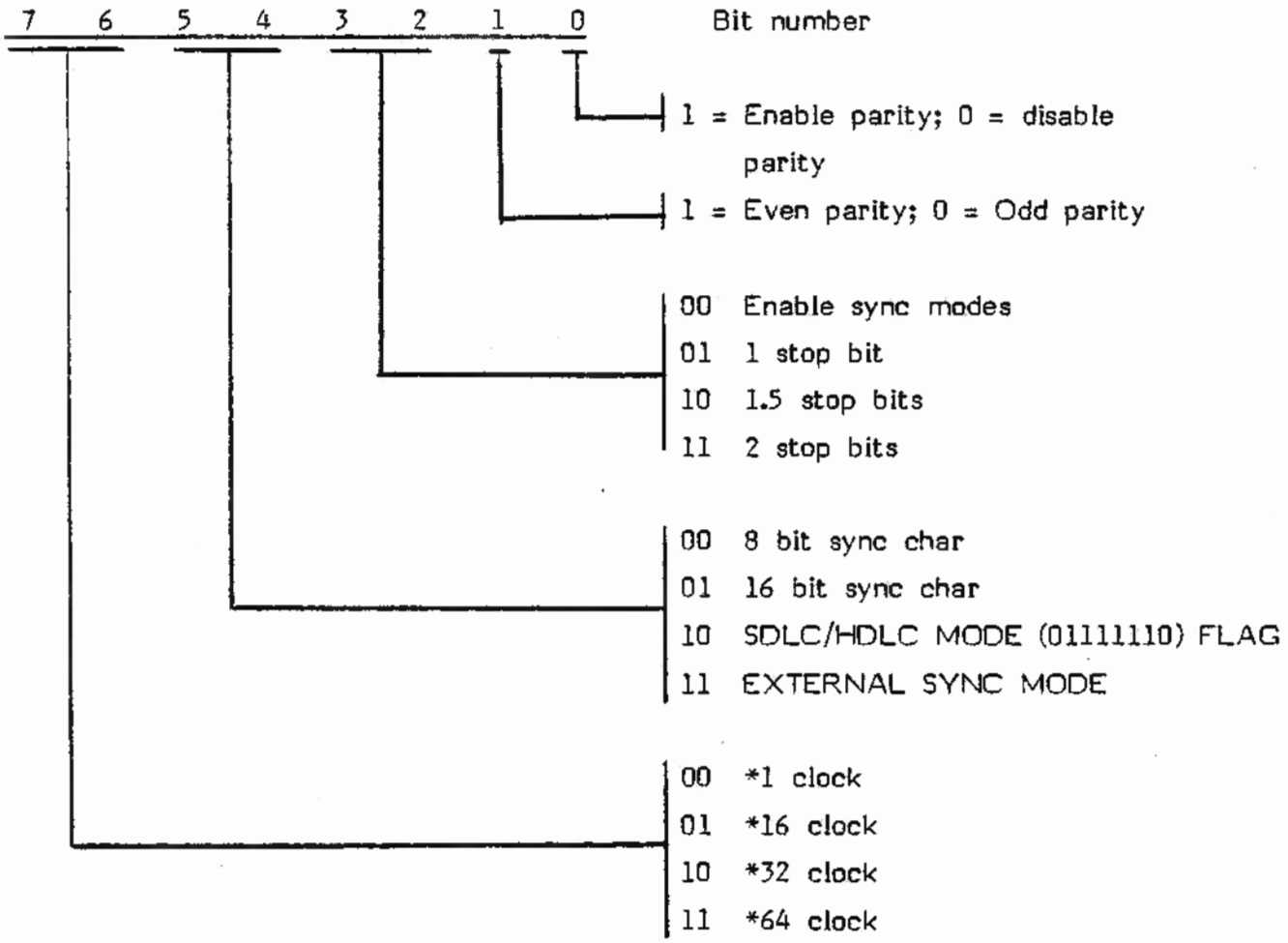
0 0 Receive 5 Data bits/character

0 1 Receive 7 Data bits/character

1 0 Receive 6 Data bits/character

1 1 Receive 8 Data bits/character

WRITE REGISTER 4 (WR4)



D0 Parity - a one in this bit causes a parity bit to be added to the programmed number of data bits per character for both the transmitted and received character. If the MPSC is programmed to receive 8 bits per character, the parity bit is not transferred to the microprocessor. With other receiver character lengths, the parity bit is transferred to the mocroprocessor.

D1 Even/Odd Parity - if parity is enabled, a one in this bit cuases the MPSC to transmit and expect even parity, and a zero causes it to send and expect odd parity.

D3,D2 Stop bits/sync mode.

0 0 Selects synchronous modes.

0 1 Async mode, 1 stop bit/character

1 0 Async mode, 1 - 1/2 stop bits/character

1 1 Async mode, 2 stop bits/character

D5,D4 Sync mode select

0 0 8 bit cyns character

0 1 16 bit sync character

1 0 SDLC mode (Flag sync)

1 1 External sync mode

D7,D6 Clock mode- selects the clock/data rate multiplier for both the receiver and the transmotter. 1x mode must be selected for synchronous modes. If the 1x mode is selected, bit synchronous must be done externally.

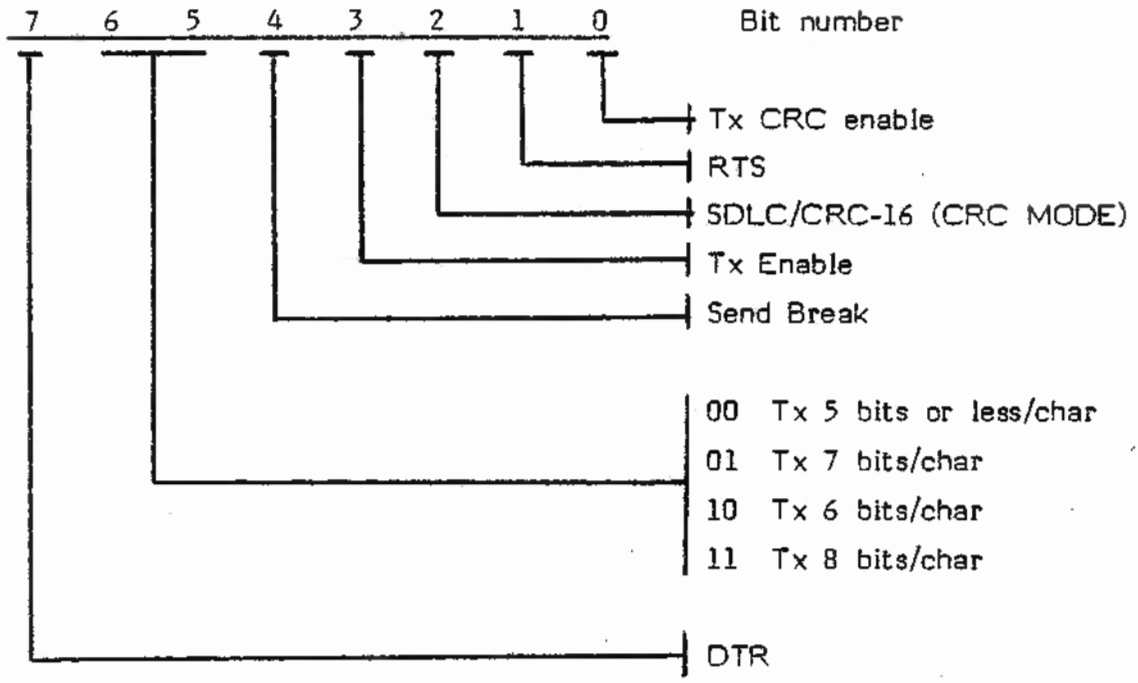
0 0 Clock rate = Data rate x 1

0 1 Clock rate = Data rate x 16

1 0 Clock rate = Data rate x 32

1 1 Clock rate = Data rate x64

WRITE REGISTER 5 (WR5):



D0 Transmit CRC Enable - a one in this bit enables the transmitter CRC generator. The CRC calculation is done when a character is moved from the transmit buffer into the shift register. A zero in this bit disables CRC calculations. If this bit is not set when a transmitter underrun occurs, the CRC will not e sent.

D1 Request to Send - a one in this bit forces the RTS pin active (low) and zero in this bit forces the RTS pin inactive (high)

D2 CRC Select - a one in this bit selects the CRC - 16 polynomial ($x^{16} + x^{15} + x^{2} + 1$) and a zero in this bit selects the CCITT-CRC polynomial ($x^{16} + x^{12} + x^{5} + 1$).

D3 Transmitter Enable - a zero in this bit forces a marking state on the transmitter output. If this bit is set to zero during data or sync character transmission, the marking state is entered after the character has been sent. If this bit is set to zero during transmission of a CRC character, sync or flag bits are substituted for the remainder of the CRC bits.

D4 Send Break - a one in this bit forces the transmit data low. A zero in this bit allows normal transmitter operation.

D6,D5 Transmit Character length

0 0 Transmit 5 or less bits/character

0 1 Transmit 7 bits/character

1 0 Transmit 6 bits/character

1 1 Transmit 8 bits/character

Bits to be sent must be right justified least significant bit first, eg:

D7	D6	D5	D4	D3	D2	D1	D0
0	0	B5	B4	B3	B2	B1	B0

Five or less mode allows transmission of one to five bits per characte. The microprocessor must format the data in the following way:

D7	D6	D5	D4	D3	D2	D1	D0	
1	1	1	1	0	0	0	B0	Sends one data bit
1	1	1	0	0	0	B1	B0	Sends two data bits
1	1	0	0	0	B2	B1	B0	Sends three data bits
1	0	0	0	B3	B2	B1	B0	Sends four data bits
0	0	0	B4	B3	B2	B1	B0	Sends five data bits

D7 Data Terminal Ready - This function is not used in MicroMikko, see 8-bit dataregister.

WRITE REGISTER 6 (WR6):

MSB							LSB
D7	D6	D5	D4	D3	D2	D1	D0

Least significant
Sync byte (Address in
SDLC/HDLC Mode)

D7-D0 Sync/Address - this register contains the transmit sync character in Monosync mode, the low order 8 sync bits in bisync mode, or the Address byte in SDLC mode.

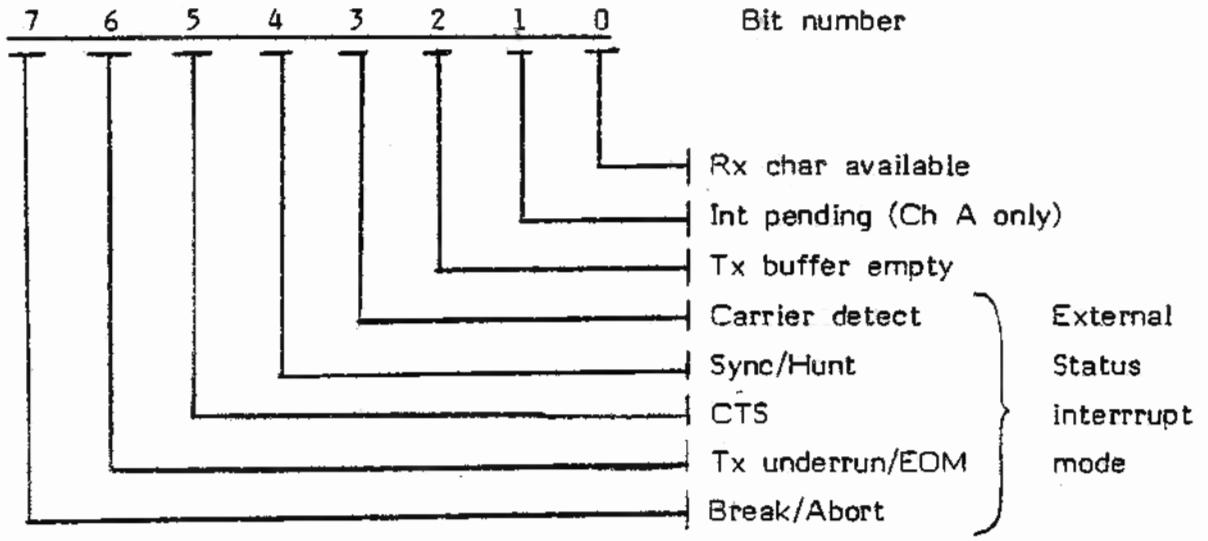
WRITE REGISTER 7 (WR7):

MSB							LSB
D7	D6	D5	D4	D3	D2	D1	D0

Most Significant
Sync byte (must be
01111110 in SDLC/HDLC
Mode)

D7-D0 Sync/Flag - this register contains the receive sync character in Monosync mode, the high order 8 sync bits in Bisync mode, or the Flag character (01111110) in SDLC mode. WR7 is not used in External Sync mode.

READ REGISTER 0 (RR0):



D0 Receive Character Available - this bit is set when the receive FIFO contains data and is reset when the FIFO is empty.

D1 Interrupt Pending* - This Interrupt-Pending bit is reset when an EOI command is isuued and there is no other interrupt request pending at that time.

D2 Transmit Buffer Empty - This bit is set whenever the transmit buffer is empty except when CRC characters are being sent in a synchronous mode. This bit is reset when the transmit buffer is loaded. This bit is set after an MPSC reset.

D3 Carrier Detect - This bit contains the state of the CD pin at the time of the last change of any of the External/Status bits (CD, CTS, Sync/Hunt, Break/Abort, or Tx Underrun/EOM). Any change of state of the CD pin causes the CD bit to be latched and causes an External/Status interrupt. This bit indicates current state of the CD pin immediately following a Reset External/Status Interrupt command.

* In vector mode this bit is set at the falling edge of the second INTA in an INTA cycle for an internal interrupt request. In non-vector mode, this bit is set at the falling edge of RD input after pointer 2 is specified. This bit is always zero in Channel B.

D4 Sync/Hunt - In asynchronous modes, the operation of this bit is similar to the CD status bit, except that Sync/Hunt shows the state of the SYNDET input. Any High-to-Low transition on the SYNDET pin sets this bit, and causes an External/Status interrupt (if enabled). The Reset External/Status Interrupt command is issued to clear the interrupt. A Low-to-High transition clears this bit and sets the Extenal/Status interrupt. When the External/Status interrupt is set by the change in state of any other input or condition, this bit shows the inverted state of the SYNDET pin at time of the change. This bit must be read immediately following a Reset External/Status Interrupt command to read the current state of the SYNDET input.

In the External Sync mode, the Sync/Hunt bit operates in a fashion similar to the Asynchronous mode, except the Enter Hunt Mode control bit enables the external sync detection logic. When the External Sync Mode and Enter Hunt Mode bits are set (forexample, when the receiver is enabled following a reset), the SYNDET input must be held High by the external logic until external character synchronization is achieved. A High at the SYNDET input holds the Sync/Hunt status in the reset condition.

When external synchronization is achieved, SYNDET must be driven Low on the second rising edge of RxC after the rising edge or RxC on which the last bit of the sync character was received. In other words, after the sync pattern is detected, the external logic must wait for two full Receive Clock cycles to activate the SYNDET input. Once SYNDET is forced Low, it is good practice to keep it Low until the CPU informs the external sync logic that synchronization has been lost or a new message is about to start. The High-to-Low transition of the SYNDET output sets the Sync/Hunt bit which sets the External/Status interrupt. The CPU must clear the interrupt by issuing the Reset External/ Status Interrupt Command.

When the SYNDET input goes High again, another External/Status interrupt is generated that must also be cleared. The Enter Hunt Mode control bit is set whenever character synchronization is lost

or the end of message is detected. In this case, the MPSC again looks for a High-to-Low transition on the SYNDET input and the operation repeats as explained previously. This implies the CPU should also inform the external logic that character synchronization has been lost and that the MPSC is waiting for SYNDET to become active.

In the Monosync and Bisync Receive modes, the Sync/Hunt status bit is initially set to 1 by the Enter Hunt Mode bit. The Sync/Hunt bit is reset when the MPSC establishes character synchronization. The High-to-Low transition of the Sync/Hunt bit causes an External/Status interrupt that must be cleared by the CPU issuing the Reset External/Status Interrupt command. This enables the MPSC to detect the next transition of other External/Status bits.

When the CPU detects the end of message or that character synchronization is lost, it sets the Enter Hunt Mode control bit, which sets the Sync/Hunt bit to 1. The Low-to-High transition of the Sync/Hunt bit sets the External/Status Interrupt, which must also be cleared by the Reset External/Status Interrupt Command. Note that the SYNDET pin act as an output in this mode, and goes low every time a sync pattern is detected in the data stream.

In the SDLC mode, the Sync/Hunt bit is initially set by the Enter Hunt mode bit, or when the receiver is disabled. In any case, it is reset to 0 when the opening flag of the first frame is detected by the MPSC. The External/Status interrupt is also generated, and should be handled as discussed previously.

Unlike the Monosync and Bisync modes, once the Sync/Hunt bit is reset in the SDLC mode, it does not need to be set when the end of message is detected. The MPSC automatically maintains synchronization. The only way the Sync/Hunt bit can set again is by the Enter Hunt Mode bit, or by disabling the receiver.

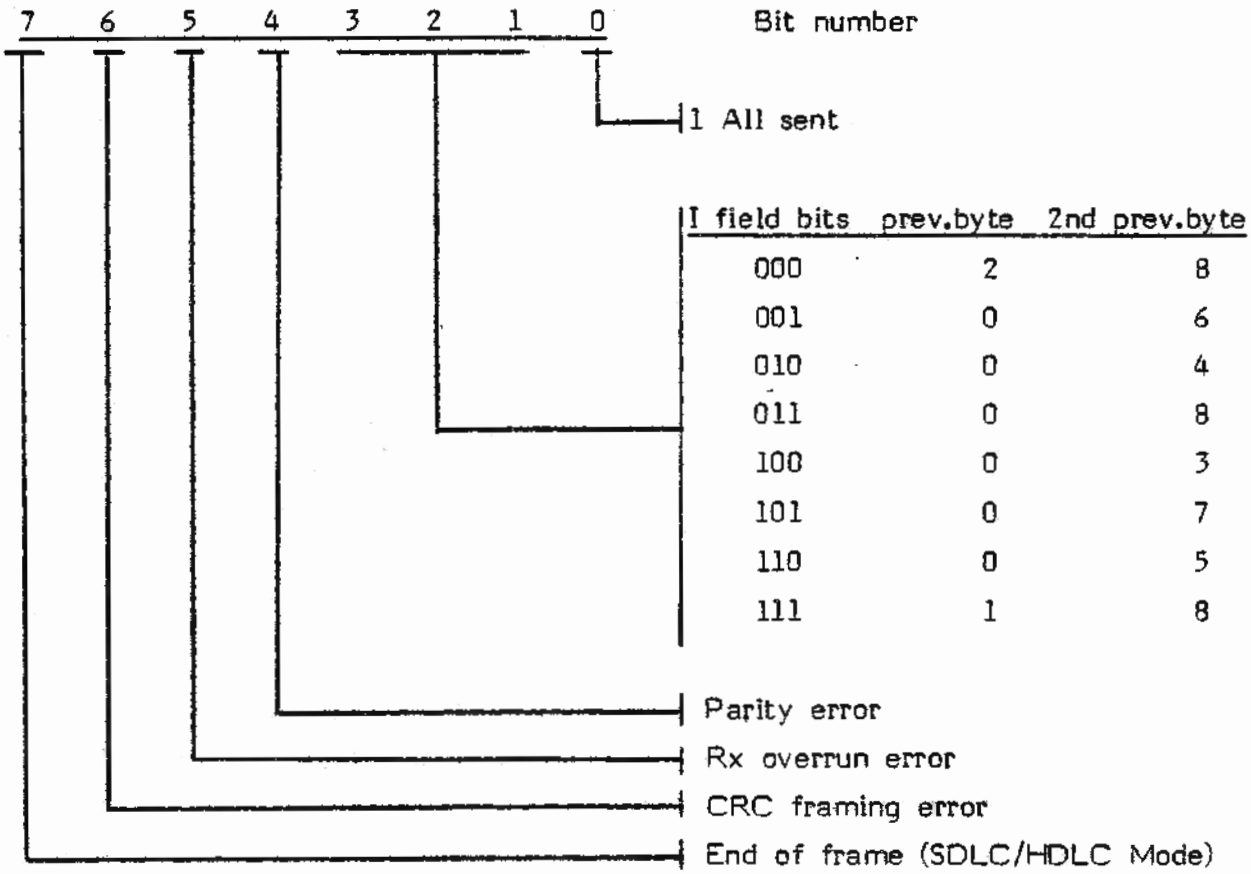
D5 Clear to Send - this bit contains the inverted state of the CTS pin at the time of the last change of any of the External/Status bits (CD, CTS, Sync/Hunt, Break/Abort, or Tx Underrun/EOM). Any

change of state of the CTS pin causes an External/Status interrupt. This bit indicates the inverse of the current state of the CTS pin immediately following a Reset External/Status Interrupt Command.

D6 Transmitter Underrun/End of Message- this bit is in a set condition following a reset (internal or external). The only command that can reset this bit is the Reset Transmit Underrun/EOM Latch command (WR0, D_6 and D_7). When the Transmit Underrun condition occurs, this bit is set, which causes the Extenal/Status Interrupt which must be reset by issuing a Reset External/Status command (WR0, command 2).

D7 Break/Abort - in the Asynchoronous Receive mode, this bit is set when a Break sequence (null character plus framing error) is detected in the data stream. The External/Status interrupt, if enabled, is set when break is detected. The interrupt service routine must issue the Reset External/Status Interrupt command (WR0, Command 2) to the break detection logic so the Break sequence termination can be recognized.

READ REGISTER 1 (RR 1)



The Break/Abort bit is reset when the termination of the Break sequence is detected in the incoming data stream. The termination of the Break sequence also causes the External/Status interrupt to be set. The Reset External/Status Interrupt command must be issued to enable the break detection logic to look for the next Break sequence. A single extraneous null character is present in the receiver after the termination of a break; it should be read and discarded.

In the SDLC Receive mode, this status bit is set by the etection of an Abort sequence (seven or more 1's). The External/Status interrupt is handled the same way as in the case of a Bfreak. The Break/Abort bit is not used in the Synchronous Receive mode.

SDLC Residue Code Table (I Field Bits in 2 Previous Bytes)

	8 bits/char		7 bits/char		6 bits/char		5 bits/char	
RR1 D3, D2 D1	Previous Byte	2nd Prev. Byte	Previous Byte	2nd Prev. Byte	Previous Byte	2nd Prev. Byte	Previous Byte	2nd Prev. Byte
1 0 0	0	3	0	2	0	1	0	5
0 1 0	0	4	0	3	0	2	0	1
1 1 0	0	5	0	4	0	3	0	2
0 0 1	0	6	0	5	0	4	0	3
1 0 1	0	7	0	6	0	5	—	—
0 1 1	0	8	0	—	—	—	—	—
1 1 1	1	8	—	—	—	—	—	—
0 0 0	2	8	1	7	0	6	0	4

D0 All sent- this bit is set when all characters have been sent, in asynchronous modes. It is reset when characters are in the transmitter, in asynchronous modes. In synchronous modes, this bit is always set.

D3,D2,D1 Residue Codes - bit synchronous protocol allow I-fields that are not an integral number of characters. Since transfers from the MPSC to the CPU are character oriented, the residue codes provide the capability of receiving leftover bits. Residue bits are right justified in the last two data bytes received.

D4 Parity Error - If parity is enabled, this bit is set for received characters whose parity does not match the programmed sense (Even/Odd). This bit is latched. Once an error occurs, it remains set until the Error Reset command is written.

D5 Receive Overrun Error - this bit indicates that the receive FIFO has been overloaded by the receiver. Tha last character in the FIFO is overwritten and flagged with this error. Once the overwritten character is read, this error condition is latched until reset by the Error Reset command. If the MPSC is in the status affects vector mode, the overrun caouses a special Receive Condition Vector.

D6 CRC/Framing Error - In async modes, a one in this bit indicates a receive framing error. In synchronous modes, a one in this bit indicates that the claculated CRC value does not match the last two bytes received. It can be reset by issuing an Error Reset command.

D7 End of Frame - this bit is valid only in SDLC mode. A one indicates that a valid ending flag has been received. This bit is reset either by an Error Reset command or upon reception of the first character of the next frame.

READ REGISTER 2 (RR2):

MSB							LSB
V7	V6	V5	V4	V3	V2	V1	V0

Interrupt Vector

Variable In Status Affects Vector Mode (WR1; D2)

RR2 Channel B

D7-D0 Interrupt vector - contains the interrupt vector programmed into WR2. If the status affects vector mode is selected, it contains the modified vector. (See WR2) RR2 contains the modified vector for the highest priority interrupt pending. If no interrupts are pending, the variable bits in the vector are set to one.

SYSTEM INTERFACE

General

The MPSC to Microprocessor System interface can be configured in many flexible ways. The basic interface types are polled, wait, interrupt driven, or direct memory access driven.

Polled operation is accomplished by repetitively reading the status of the MPSC, and making decisions based on that status. The MPSC can be polled at any time.

Wait operation allows slightly faster data throughput for the MPSC by manipulating the Ready input to the microprocessor. Block Read or Write Operations to the MPSC are started at will by the microprocessor and the MPSC deactivates its RDY signal if it is not yet ready to transmit the new byte, or if reception of new byte is not completed.

Interrupt driven operation is accomplished via an internal or external interrupt controller. When the MPSC requires service, it sends an interrupt request signal to the microprocessor, which responds with an interrupt acknowledge signal. When the internal or external interrupt controller receives the acknowledge, it vectors the microprocessor to a service routine, in which the transaction occurs.

DMA operation is accomplished via an external DMA controller. When the MPSC needs a data transfer, it request a DMA cycle from the DMA controller. The DMA controller then takes control of the bus and simultaneously does a read from the MPSC and a write to memory or vice-versa.

The following section describes the many configurations of these basic types of system interface techniques for both serial channels.

POLLED OPERATION:

In the polled mode, the CPU must monitor the desired conditions within the MPSC by reading the appropriate bits in the read registers. All data available, status, and error conditions are represented by the appropriate bits in read registers 0 and 1 for channels A and B.

There are two ways in which the software task of monitoring the status of the MPSC has been reduced. One is the "ORing" of all conditions into the Interrupt Pending bit. (RR0; D1 channel A only): This bit is set when the MPSC requires service, allowing the CPU to monitor one bit instead of four status registers. The other is available when the "status-affects-vector" mode is selected. By reading RR2 Channel B, the CPU can read a vector who's value will indicate that one or more of group of conditions has occured, narrowing the field of possible conditions. See WR2 and RR2 in the Detailed Command Description section.

WAIT OPERATION

Wait Operation cannot be used in MicroMikko due to that RDY pin in CPU is not connected to MPSC.

INTERRUPT DRIVEN OPERATION:

The MPSC can be programmed into several interrupt modes: Non-Vectored, 8085 vectored, and 8088/86 vectored. In both vectored modes, multiple MPSC's can be daisy-chaned.

In the vectored mode, the MPSC responds to an interrupt acknowledge sequence by placing a call instruction (8085 mode) and interrupt vector (8085 and 8088/86 mode) on the data bus. In the non-vectored mode, the MPSC does not respond to INTA sequences and thus cannot be used in MicroMikko

The MPSC can be programmed to cause an interrupt due to up to 14 conditions in each channel. The status of these interrupt conditions is contained in Read Registers 0 and 1. These 14 conditions are all directed to cause 3 different types of internal interrupt request for each channel: receive/interrupts, transmit interrupts and external/status interrupts (if enabled).

This results in up to 6 internal interrupt request signals. The priority of those signals can be programmed to one of two fixed modes:

Highest Priority					Lowest Priority
RxA	RxB	TxA	TxB	ExTA	ExTB
RxA	TxA	RxB	TxB	ExTA	ExTB

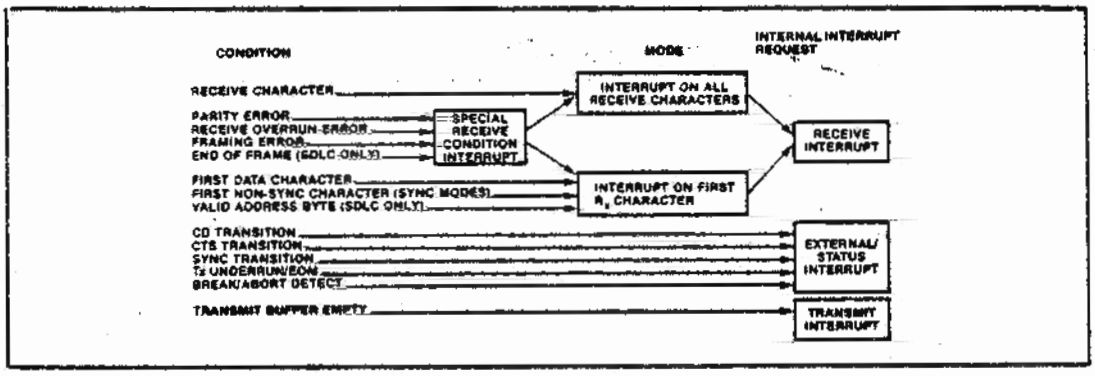
The interrupt priority resolution works differently for vectored and non-vectored modes. MPSC should be programmed to 8085 mode 1 interrupt operation.

PRIORITY RESOLUTION: VECTORED MODE

Any interrupt condition can be accepted internally to the MPSC at any time, unless the MPSC's internal INTA signal is active, unless a higher priority interrupt is currently accepted, or if IPI is inactive (high). The MPSC's internal INTA is set

on the leading (falling) edge of the first External INTA pulse and reset on the trailing (rising) edge of the second External INTA pulse. After an interrupt is accepted internally, an External INT request is generated and the IPO goes inactive. IPO and IPI are used for daisy-chaining MPSC's together.

Interrupt Conditon Grouping



The MPSC's internal INTA is set on the leading (falling) edge of the first external INTA pulse, and reset on the trailing (rising) edge of the second external INTA pulse. After an interrupt is accepted internally, an external INT request is generated.

IN-service

Each of the six interrupt sources has an associated In-Service latch. After priority has been resolved, the highest priority In-Service latch is set. After the In-Service latch is set, the INT goes inactive (high).

EOI-Command

Lower priority interrupts are not accepted internally while the In-Service latch is set. However, higher priority interrupts are accepted internally and a new external INT request is generated. If the CPU responds with a new INTA sequence, the MPSC will respond as before, suspending the lower priority interrupts.

After the interrupt is serviced, the End-of-Interrupt (EOI) command should be written to the MPSC. This command will cause an internal pulse that is used to reset the In-Service Latch which allows service for lower priority interrupts.

DMA OPERATION

MPSC can be programmed to utilize two DMA channels: Transmit Channel A, Receive Channel A. Each DMA Channel has an associated DMA Request line. Acknowledgement of a DMA cycle is done via normal data read or write cycles. This is accomplished by encoding the DACK signals to generate A_0, A_1, and CS signals, and multiplexing them with the norma A_0, A_1, and $\overline{CS}$ signals.

PERMUTATIONS

Channels A and B can be used with different system interface modes. In all cases it is possible to poll the MPSC. The following table shows the possible permutations of interrupt, wait, and DMA modes for channels A and B. Bits D_1, D_0 of WR2 Ch. A determine these permutations.

Permutation WR2 Ch. A D_1, D_0	Channel A	Channel B
0 0	Wait Interrupt Polled	Wait Interrupt Polled
0 1	DMA Polled	Interrupt Polled
1 0	DMA Polled	DMA Polled

D1, D0 = 1, 1 is illegal.
DMA for Channel B cannot be used.

Basic initialisization of MPSC in MicroMikko

Following bytes are written to control registers of MPSC

channel A:
18H = Channel Reset
02H = Pointer to WR2
21H
04H = WR4
4FH
01H = WR1
0AH
03H = WR3
61H
05H = WR5
AAH

Channel B: 18H = Channel Reset
02H = Pointer To WR2
00H
04H = WR4
4FH
01H = WR1
04H
03H = WR3
61H
05H = WR5
AAH

These select Asynchronous operation with 7 databits, two stopbits, and even parity. Clock data rate multiplier is 16, RTS = 1, Tx enable, Rx enable.

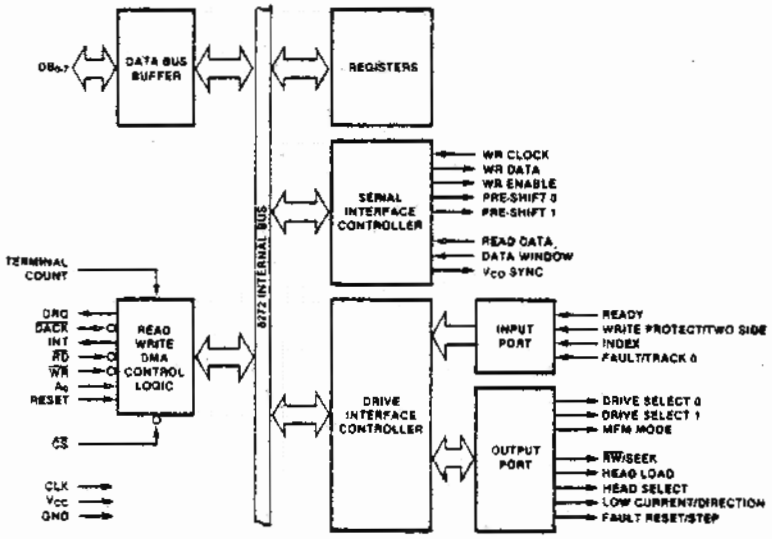
Channel A = DMA, B = polled, base int address = 0
Channel A is set to int. on first Rx character, after having got this character is DMA enabled.

DMA is prepared with disabling Rx channel, and writing input buffer address and length to DMA registers.

DTR is selected along with signal TX.21 in 8-bit dataregisters.

FLOPPY DISK CONTROL

Floppy Disk Controller is realized with Intels 8272, FDC chip or equal. This contains the circuitry and control functions for interfacing the processor to Floppy Disk Drivers. it is capable of supporting either IBM 3740 single density format (FM) or IBM system 34 double density format (MFM) included double sided recording.



DESCRIPTION

Hand-shaking signals are provided in the 8272 which make DMA operation easy to incorporate with the aid of an external DMA Controller chip, such as the 8237 used in MicroMikko. The FDC will operate in either DMA or Non-DMA mode. In the Non-DMA mode, the FDC generates interrupts to the processor for every transfer of a data byte between the CPU and the 8272. In the DMA mode, the processor need only load a command into the FDC and all data transfers occur under control of the 8272 and DMA controller. The DMA mode is normally used in Micro Mikko. Interrupt request is coupled to processors RST 5,5 pin.

There are 15 separate commands which the 8272 will execute. Each of these commands require multiple 8-bit bytes to fully specify the operation which the processor wishes the FDC to peform. The following commands are available.

Read Data	Write Data
Read ID	Format a Track
Read Deleted Data	Write Deleted Data
Read a Track	Seek
Scan Equal	Recalibrate (Restore to
Scan High or Equal	Track 0)
Scan low or Equal	Sence Interrupt Status
Specify	Sense Drive Status

8272 REGISTERS - CPU INTERFACE

The 8272 contains two registers which may be accessed by the main system processor; a Status Register and a Data Register. The 8-bit Main Status Register contains the status information of the FDC, and may be accessed at any time. The 8-bit Data Register (actually consists of several registers in a stack with only one register presented to the data bus at a time), store data commands, parameters, and FDD status information. Data bytes are read out of, or written into, the Data Register in order to program or obtain the results after execution of a command. The Status Register may only be read and is used to facilitate the transfer of data between the processor and 8272.

The relationship between the Status/Data register and READ/WRITE signals is shown below.

READ	FF50	READ MAIN STATUS REGISTER
WRITE	FF50	ILLEGAL
READ	FF51	READ FROM DATA REGISTER
WRITE	FF51	WRITE INTO DATA REGISTER

The bits in the Main Status Register are defined as follows:

BIT NUMBER	NAME	SYMBOL	DESCRIPTION
DB_0	FDD 0 Busy	D_0B	FDD number 0 is in the Seek mode.
DB_1	FDD 1 Busy	D_1B	FDD number 1 is in the Seek mode.
DB_2	FDD 2 Busy	D_2B	FDD number 2 is in the Seek mode.
DB_3	FDD 3 Busy	D_3B	FDD number 3 is in the Seek mode.
DB_4	FDC Busy	CB	A read or write command is in process.
DB_5	Non-DMA mode	NDM	The FDC is in the non-DMA mode. This bit is set only during the execution phase in non-DMA mode. Transition to "0" state indicates execution phase has ended.
DB_6	Data Input/Output	DIO	Indicates direction of data transfer between FDC and Data Register. If DIO = "1" then transfer is from Data Register to the Processor. If DIO = "0", then transfer is from the Processor to Data Register.
DB_7	Request for Master	RQM	Indicates Data Register is ready to send or receive data to or from the Processor. Both bits DIO and RQM should be used to perform the handshaking functions of "ready" and "direction" to the processor.

The DIO and ROM bits in the Status Register indicate when Data is ready and in which direction data will be transferred on the Data Bus.

The 8272 is capable of executing 15 different commands. Each command is initiated by a multi-byte transfer from the processor, and the result after execution of the command may also be a multi-byte transfer back to the processor. Because of this multi-byte interchange of information between the 8272 and the processor, it is convenient to consider each command as consisting of three phases:

Command Phase: The FDC receives all information required to perform a particular operation from the processor.

Execution Phase: The FDC performs the operation it was instructed to do.

Result Phase: After completion of the operation, status and other housekeeping information are made available to the processor.

During Command or Result Phases the Main Status Register (described earlier) must be read by the processor before each byte of information is written into or read from the Data Register. Bits D6 and D7 in the Main Status Register must be in a 0 and 1 state, respectively, before each byte of the command word may be written into the 8272. Many of the commands require multiple bytes, and as a result the Main Status Register must be read prior to each byte transfer to the 8272. On the other hand, during the Result Phase, D6 and D7 in the Main Status Register must both be 1's (D6=1 and D7=1) before reading each byte from the Data Register. Note, this reading of the Main Status Register before each byte transfer to the 8272 is required in only the Command and Result Phases, and NOT during the Execution Phase.

During the Execution Phase, the Main Status Register need not be read. If the 8272 is in the Non-DMA Mode then the receipt of each data byte (if 8272 is reading data from FDD) is indicated by an Interrupt signal. The generation of a Read signal will reset the Interrupt as well as output the Data onto the Data Bus. For example, if the processor cannot handle Interrupts fast enough (every 13 us for MFM mode) then it may poll the Main Status Register and then bit D7 (ROM) functions just like the Interrupt signal. If a Write Command is in process then the WR signal performs the reset to the Interrupt signal.

In the 8272 in the DMA Mode, no Interrupts are generated during the Execution Phase. The 8272 generates DRQ's (DMA Requests) when each byte of data is available. The DMA Controller responds to this request with both a DMA Acknowledge and a Read signal. When the DMA Acknowledge signal goes low then the DMA Request is reset. If a Write Command has been programmed then a WR signal will appear instead of RD. After the Execution Phase has been completed (Terminal Count has occured) then an Interrupt will occur. This signifies the beginning of the Result Phase. When the first byte of data is read during the Result Phase, the Interrupt is automatically reset.

It is important to note that during the Result Phase all bytes shown in the Command Table must be read. The Read Data Command, for example, has seven bytes of data in the Result Phase. All seven bytes must be read in order to

successfully complete the Read Data Command. The 8272 will not accept a new command until all seven bytes have been read. Other commands may require fewer bytes to be read during the Result Phase.

The 8272 contains five Status Register. The Main Status Register mentioned above may be read by the processor at any time. The other four Status Registers (ST0, ST1, ST2, and ST3) are only available during the Result Phase, and may be read only after successfully completing a command. The particular command which has been executed determines how many of the Status Registers will be read.

The bytes of data which are sent to the 8272 to form the Command Phase, and are read out of the 8272 in the Result Phase, must occur in the order shown in the Command Table. That is, the Command Code must be sent first and the other bytes sent in the prescribed sequence. No foreshortening of the Command or Result Phases are allowed. After the last byte of data in the Command Phase is sent to the 8272 the Execution Phase automatically starts. In a similar fashion, when the last byte of data is read out in the Result Phase, the command is automatically ended and the 8272 is ready for a new command. A command may be aborted by simply sending a Terminal Count signal. This is a convenient means of ensuring that the processor may always get the 8272's attention even if the disk system hangs up in an abnormal manner.

POLLING FEATURE OF THE 8272

After the Specify command has been sent to the 8272, the Drive Select Lines DS0 and DS1 will automatically go into a polling mode. In between commands (and between step pulses in the SEEK command) the 8272 polls all four FDD's looking for a change in the Ready line from any of the drives. If the Ready line changes state (usually due to a door opening or closing) then the 8272 will generate an interrupt. When Status Register 0 (ST0) is read (after Sense Interrupt Status is issued), Not Ready (NR) will be indicated. The polling of the Ready line by the 8272 occurs continously between instructions, thus notifying the processor which drives are on or off line.

8272 Command Set

PHASE	R/W	DATA BUS D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	REMARKS
READ A TRACK										
Command	W	0	MFM	SK	0	0	0	1	0	Command Codes
	W	0	0	0	0	0	HDS	DS1	DS0	
	W	C								Sector ID information prior to Command execution
	W	H								
	W	R								
	W	N								
	W	EOT								
	W	GPL								
	W	DTL								
Execution										Data transfer between the FDD and main-system. FDC reads all of cylinders contents from index hole to EOT
Result	R	ST 0								Status information after Command execution
	R	ST 1								
	R	ST 2								
	R	C								Sector ID information after Command execution
	R	H								
	R	R								
	R	N								
READ ID										
Command	W	0	MFM	0	0	1	0	1	0	Commands
	W	0	0	0	0	0	HDS	DS1	DS0	
Execution										The first correct ID information on the Cylinder is stored in Data Register
Result	R	ST 0								Status information after Command execution
	R	ST 1								
	R	ST 2								
	R	C								Sector ID information during Execution Phase
	R	H								
	R	R								
	R	N								
FORMAT A TRACK										
Command	W	0	MFM	0	0	1	1	0	1	Command Codes
	W	0	0	0	0	0	HDS	DS1	DS0	
	W	N								Bytes/Sector
	W	SC								Sectors/Cylinder
	W	GPL								Gap 3
	W	D								Filler Byte
Execution										FDC formats an entire cylinder
Result	R	ST 0								Status information after Command execution
	R	ST 1								
	R	ST 2								
	R	C								In this case, the ID information has no meaning
	R	H								
	R	R								
	R	N								
SCAN EQUAL										
Command	W	MT	MFM	SK	1	0	0	0	1	Command Codes
	W	0	0	0	0	0	HDS	DS1	DS0	
	W	C								Sector ID information prior to Command execution
	W	H								
	W	R								
	W	N								
	W	EOT								
	W	GPL								
	W	STP								
Execution										Data compared between the FDD and main-system
Result	R	ST 0								Status information after Command execution
	R	ST 1								
	R	ST 2								
	R	C								Sector ID information after Command execution
	R	H								
	R	R								
	R	N								

PHASE	R/W	DATA BUS D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	REMARKS
SCAN LOW OR EQUAL										
Command	W	MT	MFM	SK	1	1	0	0	1	Command Codes
	W	0	0	0	0	0	HDS	DS1	DS0	
	W	C								Sector ID information prior Command execution
	W	H								
	W	R								
	W	N								
	W	EOT								
	W	GPL								
	W	STP								
Execution										Data compared between the FDD and main-system
Result	R	ST 0								Status information after Command execution
	R	ST 1								
	R	ST 2								
	R	C								Sector ID information after Command execution
	R	H								
	R	R								
	R	N								
SCAN HIGH OR EQUAL										
Command	W	MT	MFM	SK	1	1	1	0	1	Command Codes
	W	0	0	0	0	0	HDS	DS1	DS0	
	W	C								Sector ID information prior Command execution
	W	H								
	W	R								
	W	N								
	W	EOT								
	W	GPL								
	W	STP								
Execution										Data compared between the FDD and main-system
Result	R	ST 0								Status information after Command execution
	R	ST 1								
	R	ST 2								
	R	C								Sector ID information after Command execution
	R	H								
	R	R								
	R	N								
RECALIBRATE										
Command	W	0	0	0	0	0	1	1	1	Command Codes
	W	0	0	0	0	0	0	DS1	DS0	
Execution										Head retracted to Track 0
SENSE INTERRUPT STATUS										
Command	W	0	0	0	0	1	0	0	0	Command Codes
Result	R	ST 0								Status information at the end of each seek operation about the FDC
	R	PCN								
SPECIFY										
Command	W	0	0	0	0	0	0	1	1	Command Codes
	W	SRT					HUT			
	W	HLT							ND	
SENSE DRIVE STATUS										
Command	W	0	0	0	0	0	1	0	0	Command Codes
	W	0	0	0	0	0	HDS	DS1	DS0	
Result	R	ST 3								Status information about FDD
SEEK										
Command	W	0	0	0	0	1	1	1	1	Command Codes
	W	0	0	0	0	0	HDS	DS1	DS0	
	W	NCN								
Execution										Head is positioned over proper Cylinder on Diskette
INVALID										
Command	W	Invalid Codes								Invalid Command Codes (NoOp — FDC goes into Standby State)
Result	R	ST 0								ST 0 = 80 (16)

8272 Command Set

PHASE	R/W	DATA BUS D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	REMARKS
READ DATA										
Command	W	MT	MFM	SK	0	0	1	1	0	Command Codes
	W	0	0	0	0	0	HDS	DS1	DS0	
	W	C								Sector ID information prior to Command execution
	W	H								
	W	R								
	W	N								
	W	EOT								
	W	GPL								
	W	DTL								
Execution										Data transfer between the FDD and main-system
Result	R	ST 0								Status information after Command execution
	R	ST 1								
	R	ST 2								
	R	C								Sector ID information after command execution
	R	H								
	R	R								
	R	N								
READ DELETED DATA										
Command	W	MT	MFM	SK	0	1	1	0	0	Command Codes
	W	0	0	0	0	0	HDS	DS1	DS0	
	W	C								Sector ID information prior to Command execution
	W	H								
	W	R								
	W	N								
	W	EOT								
	W	GPL								
	W	DTL								
Execution										Data transfer between the FDD and main-system
Result	R	ST 0								Status information after Command execution
	R	ST 1								
	R	ST 2								
	R	C								Sector ID information after Command execution
	R	H								
	R	R								
	R	N								

PHASE	R/W	DATA BUS D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	REMARKS
WRITE DATA										
Command	W	MT	MFM	0	0	0	1	0	1	Command Codes
	W	0	0	0	0	0	HDS	DS1	DS0	
	W	C								Sector ID information prior to Command execution
	W	H								
	W	R								
	W	N								
	W	EOT								
	W	GPL								
	W	DTL								
Execution										Data transfer between the main-system and FDD
Result	R	ST 0								Status information after Command execution
	R	ST 1								
	R	ST 2								
	R	C								Sector ID information after Command execution
	R	H								
	R	R								
	R	N								
WRITE DELETED DATA										
Command	W	MT	MFM	0	0	1	0	0	1	Command Codes
	W	0	0	0	0	0	HDS	DS1	DS0	
	W	C								Sector ID information prior to Command execution
	W	H								
	W	R								
	W	N								
	W	EOT								
	W	GPL								
	W	DTL								
Execution										Data transfer between the FDD and main-system
Result	R	ST 0								Status information after Command execution
	R	ST 1								
	R	ST 2								
	R	C								Sector ID information after Command execution
	R	H								
	R	R								
	R	N								

Note: 1. Symbols used in this table are described at the end of this section.

2. $A_0 = 1$ for all operations.

3. X = Don't care, usually made to equal binary 0.

Command Mnemonics

SYMBOL	NAME	DESCRIPTION
A_0	Address Line 0	A_0 controls selection of Main Status Register ($A_0 = 0$) or Data Register ($A_0 = 1$).
C	Cylinder Number	C stands for the current selected Cylinder track number 0 through 76 of the medium.
D	Data	D stands for the data pattern which is going to be written into a Sector.
D_7-D_0	Data Bus	8-bit Data Bus where D_7 is the most significant bit, and D_0 is the least significant bit.
DS0, DS1	Drive Select	DS stands for a selected drive number 0 or 1.
DTL	Data Length	When N is defined as 00, DTL stands for the data length which users are going to read out or write into the Sector.
EOT	End of Track	EOT stands for the final Sector number of a Cylinder.
GPL	Gap Length	GPL stands for the length of Gap 3 (spacing between Sectors excluding VCO Sync Field).
H	Head Address	H stands for head number 0 or 1, as specified in ID field.
HDS	Head Select	HDS stands for a selected head number 0 or 1 (H = HDS in all command words).
HLT	Head Load Time	HLT stands for the head load time in the FDD (2 to 254 ms in 2 ms increments).
HUT	Head Unload Time	HUT stands for the head unload time after a read or write operation has occurred (16 to 240 ms in 16 ms increments).
MFM	FM or MFM Mode	If MF is low, FM mode is selected and if it is high, MFM mode is selected.
MT	Multi-Track	If MT is high, a multi-track operation is to be performed (a cylinder under both HD0 and HD1 will be read or written).
N	Number	N stands for the number of data bytes written in a Sector.

SYMBOL	NAME	DESCRIPTION
NCN	New Cylinder Number	NCN stands for a new Cylinder number, which is going to be reached as a result of the Seek operation. Desired position of Head.
ND	Non-DMA Mode	ND stands for operation in the Non-DMA Mode.
PCN	Present Cylinder Number	PCN stands for the Cylinder number at the completion of SENSE INTERRUPT STATUS Command. Position of Head at present time.
R	Record	R stands for the Sector number, which will be read or written.
R/W	Read/Write	R/W stands for either Read (R) or Write (W) signal.
SC	Sector	SC indicates the number of Sectors per Cylinder.
SK	Skip	SK stands for Skip Deleted Data Address Mark.
SRT	Step Rate Time	SRT stands for the Stepping Rate for the FDD (1 to 16 ms in 1 ms increments). The same Stepping Rate applies to all drives (F = 1 ms, E = 2 ms, etc.).
ST 0 ST 1 ST 2 ST 3	Status 0 Status 1 Status 2 Status 3	ST 0-3 stand for one of four registers which store the status information after a command has been executed. This information is available during the result phase after command execution. These registers should not be confused with the main status register (selected by $A_0 = 0$). ST 0-3 may be read only after a command has been executed and contain information relevant to that particular command.
STP		During a Scan operation, if STP = 1, the data in contiguous sectors is compared byte by byte with data sent from the processor (or DMA), and if STP = 2, then alternate sectors are read and compared.

COMMAND DESCRIPTIONS

During the Command Phase, the Main Status Register must be polled by the CPU before each byte is written into the Data Register. The DIO (DB6) and RQM (DB7) bits in the Main Status Register must be in the "0" and "1" states respectibely, before each byte of the command may be written into the 8272. The beginning of the execution phase for any of these commands will cause DIO and RQM to switch to "1" and "0" states respectively

READ DATA

A set of nine (9) byte words are required to place the FDC into the Read Data Mode. After the Read Data command has been issued the FDC loads the head (if it is in the unloaded state), waits the specified head settling time (defined in the Specify Command), and begins reading ID Address Marks and ID fields. When the current sector number ("R") stored in the ID Register (IDR) compares with the sector number off the diskette, then the FDC outputs data (from the data field) byte-by-byte to the main system via the data bus.

After completion of the read operation from the current sector, the Sector Number is incremented by one, and the data from the next sector is read and output on the data bus. This continuous read function is called a " Multi-Sector Read Operation". The Read Data Command may be terminated by the receipt of a Terminal Count signal. Upon receipt of this signal, the FDC stops outputting data to the processor, but will continue to read data from the current sector, check CRC (Cyclic Redundancy Count) bytes, and then at the end of the sector terminate the Read Data Command.

The amount of data which can be handled with single command to the FDC depends upon MT (multi-track), MFM (MFM/FM), and N (Number of Bytes/Sector). Table below shows the Transfer Capacity.

Multi-Track MT	MFM/FM MFM	Bytes/Sector N	Maximum Transfer Capacity (Bytes/Sector) (Number of Sectors)	Final Sector Read from Diskette
0	0	00	(128) (26) = 3,328	26 at Side 0
0	1	01	(256) (26) = 6,656	or 26 at Side 1
1	0	00	(128) (52) = 6,656	26 at Side 1
1	1	01	(256) (52) = 13,312	
0	0	01	(256) (15) = 3,840	15 at Side 0
0	1	02	(512) (15) = 7,680	or 15 at Side 1
1	0	01	(256) (30) = 7,680	15 at Side 1
1	1	02	(512) (30) = 15,360	
0	0	02	(512) (8) = 4,096	8 at Side 0
0	1	03	(1024) (8) = 8,192	or 8 at Side 1
1	0	02	(512) (16) = 8,192	8 at Side 1
1	1	03	(1024) (16) = 16,384	

The "multi-track" function (MT) allows the FDC to read data from both sides of the diskette. For a particular cylinder, data will be transferred starting at Sector 0, Side 0 and completing at Sector L, Side 1 (Sector L = last sector on the side). Note, this function pertains to only one cylinder (the same track) on each side of the diskette.

When N=0, then DTL defines the data length which the FDC must treat as a sector. If DTL is smaller than the actual data length in a Sector, the data beyond DTL in the Sector, is not sent to the Data Bus. The FDC reads (internally) the complete Sector performing the CRC check, and depending upon the manner of command termination, may perform a Multi-Sector Read Operation. When N is non-zero, then DTL has no meaning and should be set to 0FFH.

At the completion of the Read Data Command, the head is not unloaded until after Head Unload Time Interval (specified in the Specify Command) has elapsed. If the processor issues another command before the head unloads then the head settling time may be saved between subsequent reads. This time out is particularly variable when a diskette is copied from one drive to another.

If the FDC detects the Index Hole twice without finding the right sector, (indicated in "R"), then the FDC sets the ND (No Data) flag in Status Register 1 to a 1 (high), and terminates the Read Data Command. (Status Register 0 also has bits 7 and 6 set to 0 and 1 respectively.)

After reading the ID and Data Fields in each sector, the FDC checks the CRC bytes. If a read error is detected (incorrect CRC in ID field), the FDC sets the DE (Data Error) flag in Status Register 1 to a 1 (high), and if a CRC error occurs in the Data Field the FDC also sets the DD (Data Error in Data Field) flag in Status Register 2 to a 1 (high), and terminates the Read Data Command. (Status Register 0 also has bits 7 and 6 set to 0 and 1 respectively.)

If the FDC reads a Deleted Data Address Mark off the diskette, and the SK bit (bit D5 in the first Command Word) is not set (SK=0), then the FDC sets the CM (Control Mark) flag in Status Register 2 to a 1 (high), and terminates the Read Data Command, after reading all the data in the Sector. If SK=1, the FDC skips the sector with the Deleted Data Address Mark and reads the next sector.

During disk data transfers between the FDC and the processor, via the data bus, the FDC must be serviced by the processor every 27 us in the FM Mode, and every 13 us in the MFM Mode, or the FDC sets the OR (Over Run) flag in Status Register 1 to a 1 (high), and terminates the Read Data Command.

If the processor terminates a read (or write) operation in the FDC, then the ID Information in the Result Phase is dependent upon the state of the MT bit and EOT byte. Table shows the values for C, H, R, and N, when the processor terminates the Command.

ID INFORMATION WHEN PROCESSOR TERMINATES COMMAND

MT	EOT	Final Sector Transferred to Processor	ID Information at Result Phase			
			C	H	R	N
0	1A	Sector 1 to 25 at Side 0	NC	NC	R+1	NC
	0F	Sector 1 to 14 at Side 0				
	08	Sector 1 to 7 at Side 0				
	1A	Sector 26 at Side 0	C+1	NC	R=01	NC
	0F	Sector 15 at Side 0				
	08	Sector 8 at Side 0				
	1A	Sector 1 to 25 at Side 1	NC	NC	R+1	NC
	0F	Sector 1 to 14 at Side 1				
	08	Sector 1 to 7 at Side 1				
	1A	Sector 26 at Side 1	C+1	NC	R=01	NC
	0F	Sector 15 at Side 1				
	08	Sector 8 at Side 1				
1	1A	Sector 1 to 25 at Side 0	NC	NC	R+1	NC
	0F	Sector 1 to 14 at Side 0				
	08	Sector 1 to 7 at Side 0				
	1A	Sector 26 at Side 0	NC	LSB	R=01	NC
	0F	Sector 15 at Side 0				
	08	Sector 8 at Side 0				
	1A	Sector 1 to 25 at Side 1	NC	NC	R+1	NC
	0F	Sector 1 to 14 at Side 1				
	08	Sector 1 to 7 at Side 1				
	1A	Sector 26 at Side 1	C+1	LSB	R=01	NC
	0F	Sector 15 at Side 1				
	08	Sector 8 at Side 1				

Notes: 1. NC (No Change): The same value as the one at the beginning of command execution.

2. LSB (Least Significant Bit): The least significant bit of H is complemented.

WRITE DATA

A set of nine (9) bytes are required to set the FDC into the Write Data mode. After the Write Data command has been issued the FDC loads the head (if it is in the unloaded state), waits the specified head settling time (defined in the Specify Command), and begins reading ID Fields. When the current sector number ("R"), stored in the ID Register (IDR) compares with the sector number read off the diskette, then the FDC takes data from the processor byte-by-byte via the data bus, and outputs it to the FDD.

After writing data into the current sector, the Sector Number stored in "R" is incremented by one, and the next data field is written into. The FDC continues this "Multi-Sector Write Operation" until the issuance of a Terminal Count signal. If a Terminal Count signal is sent to the FDC it continues writing into the current sector to complete the data field. If the Terminal Count signal is received while a data field is being written then the remainder of the data field is filled with 00 (zeros).

The FDC reads the ID field of each sector and checks the CRC bytes. If the FDC detects a read error (incorrect CRC) in one of the ID Fields, it sets the DE (Data Error) flag of Status Register 1 to a 1 (high), and terminates the Write Data Command. (Status Register 0 also has bits 7 and 6 set to 0 and 1 respectively.)

The Write Command operates in much the same manner as the Read Command. The following items are the same; refer to the Read Data Command for details:

* Transfer Capacity
* EN (End of Cylinder) Flag
* ND (No Data) Flag
* Head Unload Time Interval
* ID Information when the processor terminates comman
* Definition of DTL when N=0 and when N≠0

In the Write Data mode, data transfers between the processor and FDC must occur every 31 us in the FM mode, and every 15 us in the MFM mode. If the time interval between data transfers is longer than this then the FDC sets the OR (Over Run) flag in Status Register 1 to a 1 (high), and terminates the Write Data Command.

WRITE DELETED DATA

This command is the same as the Write Data Command except Deleted Data Address Mark is written at the beginning of the Data Field instead of the normal Data Address Mark.

READ DELETED DATA

This command is the same as the Read Data Command except that when the FDC detects a Data Address Mark at the beginning of a Data Field (and SK=0 (low)), it will read all the data in the sector and set the CM flag in Status Register 2 to a 1 (high), and then terminate the command. If SK=1, then the FDC skips the sector with tha Data Address Mark and reads the next sector.

READ A TRACK

This command is similar to READ DATA Command except that the entire data field is read continuously from each of the sectors of a track. Immediately after encountering the INDEX HOLE, the FDC starts reading all data fields on the track as continuous blocks of data. If the FDC finds an error in the ID or DATA CRC check bytes, it continues to read data from the track. The FDC compares the ID information read from each sector with the value stored in the IDR, and sets the ND flag of Status Register 1 to a 1 (high) if there is no comparison. Multi-track or skip operations are not allowed with this command.

This command terminates when EOT number of sectors have been read. If the FDC does not find an ID Address Mark on the diskette after it encounters the INDEX HOLE for the second time, then it sets the MA (missing address mark) flag in Status Register 1 to a 1 (high), and terminates the command. (Status Register 0 has bits 7 and 6 set to 0 and 1 respectively.)

READ ID

The READ ID Command is used to give the present position of the recording head. The FDC stores the values from the first ID Field it is able to read. If no proper ID Address Mark is found on the diskette, before the INDEX HOLE is encountered for the second time then the MA (Missing Address Mark) flag in Status Register 1 is set to a 1 (high), and if no data is found then the ND (No Data) flag is also set in Status Register 1 to a 1 (high) and the command is terminated.

FORMAT A TRACK

The Format Command allows an entire track to be formatted. After the INDEX HOLE is detected, Data is written on the Diskette: Gaps, Address Marks, ID Fields and Data Fields, all per the IBM System 34 (Double Density) or System 3740 (Single Density) Format are recorded. The particular format which will be written is controlled by the values programmed into N (numbers of bytes/sectors) , SC (sector/cylinder), GPL (Gap Length), and D (Data Pattern) which are supplied by the processor during the Command Phase. The Data Field is filled with the Byte of data stored in D. The ID Field for each sector is supplied by the processor; that is, four data requests per sector are made by the FDC for C (Cylinder Number), H (Head Number), R (Sector Number) and N (Number of Bytes/Sector).

This allows the diskette to be formatted with nonsequential sector numbers, if desired.

After formatting each sector, the processor must send new values for C, H, R, and N to the 8272 for each sector on the track. The contents of the R register is incremented by one after each sector is formatted, thus, the R register contains a value of R+1 when it is read during the Result Phase. This incrementing and formatting continues for the whole track until the FDC encounters the INDEX HOLE for the second time, whereupon it terminates the command.

If a FAULT signal is received from the FDD at the end of a write operation, then the FDC sets the EC flag of Status Register 0 to a 1 (high), and terminates the command after setting bits 7 and 6 of Status Register 0 to 0 and 1 respectively. Also the loss of a READY signal at the beginning af a command execution phase causes command termination.

The relationship between N, SC, and GPL for various sector sizes:

Sector Size Relationships

FORMAT	SECTOR SIZE	N	SC	GPL[1]	GPL[2]	REMARKS
FM Mode	128 bytes/Sector	00	$1A_{(16)}$	$07_{(16)}$	$1B_{(16)}$	IBM Diskette 1
	256	01	$0F_{(16)}$	$0E_{(16)}$	$2A_{(16)}$	IBM Diskette 2
	512	02	08	$1B_{(16)}$	$3A_{(16)}$	
FM Mode	1024 bytes/Sector	03	04	—	—	
	2048	04	02	—	—	
	4096	05	01	—	—	
MFM Mode	256	01	$1A_{(16)}$	$0E_{(16)}$	$36_{(16)}$	IBM Diskette 2D
	512	02	$0F_{(16)}$	$1B_{(16)}$	$54_{(16)}$	
	1024	03	08	$35_{(16)}$	$74_{(16)}$	IBM Diskette 2D
	2048	04	04	—	—	
	4096	05	02	—	—	
	8192	06	01	—	—	

Note: 1. Suggested values of GPL in Read or Write Commands to avoid splice point between data field and ID field of contiguous sections.

2. Suggested values of GPL in format command.

SCAN COMMANDS

The SCAN Commands allow data which is being read from the diskette to be compared against data which is being supplied from the main system (Processor in NON-DMA mode, and DMA Controller in DMA mode). The FDC compares the data on a byte-by-byte basis, and looks for a sector of data which meets the conditions of $D_{FDD} = D_{Processor}$, $D_{FDD} \leq D_{Processor}$, or $D_{FDD} \geq D_{Processor}$. Ones complement arithmetic is used for comparison (FF = largest number, 00 = smallest number). After a whole sector of data is compared, if the conditions

are not met, the sector number is incremented (R + ST -> R), and the scan operation is continued. The scan operation continues until one of the following conditions occur, the conditions for scan are met (equal, low, or high), the last sector on the track is reached (EOT), or the terminal count signal is received.

If the conditions for scan are met then the FDC sets the SH (Scan Hit) flag of Status Register 2 to a 1 (high), and terminates the Scan Command. If the conditions for scan are not met between the starting sector (as specified by R) and the last sector on the cylinder (EOT), then the FDC sets the SN (Scan Not Satisfied) flag of Status Retister 2 to a 1 (high), and terminates the Scan Command. The receipt of a TERMINAL COUNT signal from the Processor or DMA Controller during the scan operation will cause the FDC to complete the comparison of the particular byte which is in process, and then to terminate the command. Table shows the status of bits SH and SN under various conditions of SCAN.

Scan Status codes

	STATUS REGISTER 2		
COMMAND	Bit 2=SN	Bit 3=SH	COMMENTS
Scan Equal	0	1	$D_{FDD} = D_{Processor}$
	1	0	$D_{FDD} \neq D_{Processor}$
Scan Low or Equal	0	1	$D_{FDD} = D_{Processor}$
	0	0	$D_{FDD} < D_{Processor}$
	1	0	$D_{FDD} \not< D_{Processor}$
Scan High or Equal	0	1	$D_{FDD} = D_{Processor}$
	0	0	$D_{FDD} > D_{Processor}$
	1	0	$D_{FDD} \not> D_{Processor}$

If the FDC encounters a Deleted Data Address Mark on one of the sectors (and SK=0), then it regards the sector as the last sector on the cylinder, sets CM (Control Mark) flag of Status Register 2 to a 1 (high) and terminates the command. If SK=1, the FDC skips the sector with the Deleted Address Mark, and reads the next sector. In the second case (SK=1), the FDC sets the CM (Control Mark) flag of Status Register 2 to a 1 (high) in order to show that a Deleted Sector had been encountered.

When either the STP (contiguous sectors STP=01, or alternate sectors STP=02 sectors are read) or the MT (Multi-Track) are programmed, it is necessary to remember that the last sector on the track must be read. For example, if STP=02, MT=0, the sectors are numbered sequentially 1 through 26, and we start the Scan Command at sector 21; the following will happen. Sectors 21, 23, and 25 will be read, then the next sector (26) will be skipped and the Index Hole will be encountered before the EOT value of 26 can be read. This will result in an abnormal termination of the command. If the EOT had been set at 25 or the scanning started at sector 20, then the Scan Command would be completed in a normal manner.

During the Scan Command data is supplied by either the processor or DMA Controller for comparison against the data read from the diskette. In order to avoid having the OR (Over Run) flag set in Status Register 1, it is necessary to have the data available in less than 27 us (FM Mode) or 13 us (MFM Mode). If an Overrun occurs the FDC terminates the command.

SEEK

The read/write head within the FDD is moved from cylinder to cylinder under control of the Seek Command. The FDC compares the PCN (Present Cylinder Number) which is the current head position with the NCN (New Cylinder Number), and performs the following operation if there is a difference:

PCN < NCN: Direction signal to FDD set to a 1 (high), and Step Pulses are issued. (Step In.)
PCN > NCN: Direction signal to FDD set to a 0 (low), and Step Pulses are issued . (Step Out).

The rate at which Step Pulses are issued is controlled by SRT (Stepping Rate Time) in the SPECIFY Command. After each Step Pulse is issued NCN is compared against PCN, and when NCN = PCN, then the SE (Seek End) flag is set in Status Register 0 to a 1 (high), and the command is terminated.

During the Command Phase of the Seek operation the FDC is in the FDC BUSY state, but during the Execution Phase it is in the NON BUSY state. While the FDC is in the NON BUSY state, another Seek Command may be issued, and in this manner parallel seek operations may be done on up to 4 Drives at once.

If an FDD is in a NOT READY state at the beginning of the command execution phase or during the seek operation, then the NR (NOT READY) flag is set in Status Register 0 to a 1 (high), and the command is terminated.

RECALIBRATE

This command causes the read/write head within the FDD to retract to the Track 0 position. The FDC clears the contents of the PCN counter, and checks the status of the Track 0 signal from the FDD. As long as the Track 0 signal is low, the Direction signal remains 1 (high) and Step Pulses are issued. When the Track 0 signal goes high, the SE (SEEK END) flag in Status Register 0 is set to a 1 (high) and the command is terminated. If the Track 0 signal is still low after 77 Step Pulses have been issued, the FDC sets the SE (SEEK END) and EC (EQUIPMENT CHECK) flags of Status Register 0 to both 1s (highs), and terminates the command.

The ability to overlap RECALIBRATE Commands to multiple FDDs, and the loss of the READY signal, as described in the SEEK Command, also applies to the RECALIBRATE Command.

SENSE INTERRUPT STATUS

An Interrupt signal is generated by the FDC for one of the following reasons:

1. Upon entering the Result Phase of:
   a. Read Data Command
   b. Read a Track Command
   c. Read ID Command
   d. Read Deleted Data Command
   e. Write Data Command
   f. Format a Cylinder Command
   g. Write Deleted Data Command
   h. Scan Commands
2. Ready Line of FDD changes state
3. End of Seek or Recalibrate Command
4. During Execution Phase in the NON-DMA Mode

Interrupt caused by reasons 1 and 4 above occur during normal command operations and are easily discernible by the processor. However, interrups caused by reasons 2 and 3 above may be uniquely identified with the aid of the Sense Interrupt Status Command. This command when issued resets the interrupt signal and via bits 5,6. and 7 of Status Register 0 identifies the cause of the interrupt.

SEEK, INTERRUPT CODES

SEEK END	INTERRUPT CODE		CAUSE
BIT 5	BIT 6	BIT 7	
0	1	1	Ready Line changed state, either polarity
1	0	0	Normal Termination of Seek or Recalibrate Command
1	1	0	Abnormal Termination of Seek or Recalibrate Command

Neither the Seek or Recalibrate Command have a Result Phase. Therefore, it is mandatory to use the Sense Interrupt Status Command after these commands to effectively terminate them and to provide verification of the head position (PCN).

SPECIFY

The Specify Command sets the initial values for each of the three internal timers. The HUT (Head Unload Time) defines the time from the end of the Execution Phase of one of the Read/Write Commands to the head unload state. This timer is programmable from 32 to 480 ms in increments of 32 ms (01=32 ms, 02=64 ms....OF=480 ms). The SRT (Step Rate Time) defines the time interval between adjacent step pulses. This timer is programmable from 2 to 32 ms in increments of 2 ms (F=2 ms, E=4 ms, D=6 ms, etc.). The HLT (Head Load Time) defines the time between when the Head Load signal goes high and when the Read/Write operation starts. This timer is programmable from 4 to 508 ms in increments of 4 ms (01=4 ms, 02=8 ms, 03=12 ms....FE=508 ms).

The choice of DMA or NON-DMA operation is made by the ND (NON-DMA) bit. When this bit is high (ND=1) the NON-DMA mode is selected, and when ND=0 the DMA mode is selected.

SENSE DRIVE STATUS

This command may be used by the processor whenever it wishes to obtain the status of the FDDs. Status Register 3 contains the Drive Status information.

INVALID

If an command is sent to the FDC (a command not defined above), then the FDC will terminate the command. No interrupt is generated by the 8272 during this condition. Bit 6 and bit 7 (DIO and RQM) in the Main Status Register are both high ("1") indicating to the processor that the 8272 is in the Result Phase and the contents of Status Register 0 it will find a 80H indicating an invalid command was received.

A Sence Interrupt Status Command must be sent after a Seek or Recalibrate interrupt, otherwise the FDC will consider the next command to be an Invalid Command.

In some applications the user may wish to use this command as a No-Op command, to place the FDC in a stand by or no operation state.

STATUS REGISTERS

BIT NO.	BIT NAME	BIT SYMBOL	DESCRIPTION
		STATUS REGISTER 0	
D_7	Interrupt Code	IC	D_7=0 and D_6=0 Normal termination of Command, (NT) Command was completed and properly executed.
D_6			D_7=0 and D_6=0 Abnormal Termination of Command, (AT). Execution of Command was started, but was not succesfully completed.
			D_7=1 and D_6=0 Invalid Command issue, (IC). Command which was issued was never started.
			D_7=1 and D_6=1 Abnormal Termination because during command execution the ready signal from FDD changed state.
D_5	Seek End	SE	When the FDC completes the SEEK Command, this flag is set to 1 (high).
D_4	Equipment Check	EC	If a fault Signal is received from the FDD, or if the Track 0 Signal fails to occur after 77 Step Pulses (Recalibrate Command) then this flag is set.
D_3	Not Ready	NR	When the FDD is in the not-ready state and a read or write command is issued, this flag is set. If a read or write commans is issued to Side 1 of a single sided drive, then this flag is set.

D_2	Head Address		This flag is used to indicate the state of the head at Interrupt
D_1	Unit Select1	US 1	These flags are used to indicate a Drive Unit Number at Interrupt
D_0	Unit Select0	US 0	

STATUS REGISTER 1

D_7	End of Cylinder	EN	When the FDC tries to access a Sector beyond the final Sector of a Cylinder, this flag is set.
D_6			Not used. This bit is always 0 (low).
D_5	Data Error	DE	When the FDC detects a CRC error in either the ID field or the data field, this flag is set.
D_4	Over Run	OR	If the FDC is not serviced by the main systems during data transfers, within a certain time interval, this flag is set.
D_3			Not used. This bit always 0 (low).
D_2	No Data	ND	During execution of READ DATA, WRITE DELETED DATA or SCAN Command, if the FDC cannot find the Sector specified in the IDR Register, this flag is set.
			During executing the READ ID command, if the FDC cannot read the ID field without an error, then this flag is set.
			During the execution of the READ A Cylinder Command, if the starting sector cannot be found, then this flag is set

D_1	Not Writable	NW	During execution of WRITE DATA, WRITE DELETED DATA of Format A Cylinder Command, if the FDC detects a write protect signal from the FDD, then this flag is set.
D_0	Missing Address Mark	MA	If the FDC cannot detect the ID Address Mark after encounering the index hole twice, then this flag is set.
			If the FDC cannot detect the Data Address Mark or Deleted Data Address Mark, this flag is set. Also at the same time, the MD (Missing Address Mark in Data Field) of Status Register 2 is set.

STATUS REGISTER 2

D_7			Not used. This bit is always 0 (low).
D_6	Control Mark	CM	During executing the READ DATA or SCAN Command, if the FDC encounters a Sector which contains a Deleted Data Address Mark, this flag is set.
D_5	Data Error in Data Field	DD	If the FDC detects a CRC error in the data field then this flag is set.
D_4	Wrong Cylinder	WC	This bit is related with the ND bit, and when theontents of C on the medium is different from that stored in the IDR, this flag is set.
D_3	Scan Equal Hit	SH	During execution, the SCAN Command, if the condition of "equal" is satisfied, this flag is set.

D_2	Scan Not Satisfied	SN	During executing the SCAN Command, if the FDC cannot find a Sector on the cylinder which meets the condition, then this flag is set.
D_1	Bad Cylinder	BC	This bit is related ith the ND bit, and when the content of C on the medium is different from that stored in the IDR and the content of C is FF, then this flag is set.
D_0	Missing Address Mark in Data Field	MD	When data is read from the medium, if the FDC cannot find a Data Address Mark or Deleted Data Address Mark, then this flag is set.

STATUS REGISTER 3

D_7	Fault	FT	This bit is used to indicate the status of the Fault signal from the FDD.
D_6	Write Protected	WP	This bit is used to indicate the status of the Write Protected signal from the FDD.
D_5	Ready	RDY	This bit is used to indicate the status of the REady signal from the FDD.
D_4	Track 0	T0	This bit is used to indicate the status of the Track 0 signal from the FDD.
D_3	Two Side	TS	This bit is used to indicate the status of the Two Side signal from the FDD.
D_2	Head Address	HD	This bit is used to indicate the status of Side Select signal to the FDD.
D_1	Unit Selctl	US 1	This bit is used to indicate the status of the Unit Select 1 signal to the FDD.

D_0	Unit Select0	US 0	This bit is used to indicate the status of the Unit Select 0 signal to the FDD.

THE PRACTICE HAS SHOWN FOLLOWING:

FDC goes to Busy state (Bit-4 in Main Status Register =1) when

- Doing Read or Write operation to FDD
- not enough command bytes in data register
- Ready to output result bytes after Read, Write or Sense Command

FDC sets RQM-bit to 1 when

- it has sent the interrupt
- it is ready to Transmit or Receive bytes; in DMA mode command or result bytes, in NON-DMA-mode databytes also
- it is not currently doing something else than SEEK-command (with except of NON-DMA-Mode)

The bit 0-3 in FDC main status register is "1", when

- the corresponding drive has received the complete SEEK or RECALIBRATE command.
- One of these commands above has caused an interrupt, which must be cleared with SENSE INTERRUPT STATUS command (and one resultbyte is read from the controller)

FDC - Main status register

	Main status register state	Explanation
	<80H	FDC is doing an intern operation or an I/O command via DMA. (The last one can be interrupted with TC)
DMA-MODE	80H	Basic state: Ready to take commands
	81H-8FH	Ready to take SEEK, RECALIBRATE, SDS and SIS commands. (SIS resets the SEEK-bit in lowest unit).
	90H-9FH	Command state: The command is not yet fullfilled
	A0H	NON-DMA basis state: Ready to be commanded.
	A1H-AFH	Non DMA-state like 81H-8FH
	B0H	NON DMA command state (I/O command) or request for data transfer in I/O executing phase
	B1H-BFH	NON-DMA command state (SEEK, RECALIBRATE, SDS)
	C0H-CFH	-
	D0H-DFH	Dataregister readable; result phase
	E0H-EFH	-
	F0H	NON-DMA result state (like D0) or dataread-request in I/O execution phase
	F1H-FFH	NON-DMA result state, resultbyte ready for reading in dataregister

Status register state	ST0
00H-07H	Operation succesfully completed (No SEEK or RECALIBRATE)
08H-1FH	-
20H-27H	SEEK or RECALIBRATE succesfully completed
28H-3FH	-
40H-47H	Operation not succesfull (see ST1 and ST2)
48H-4FH	Operation not succesfull, due to drive being not in READY state or the unit/head not exists.
50H-57H	The unit not in order (FAULT=1)
58H-5FH	-
60H-67H	(SEEK or RECAL not succeeded)
68H-6FH	-
70H-77H	RECAL not succesfull
78H-7FH	-
80H	Invalid command
81H-BFH	-
C0H-CFH	The READY-signal of the unit has changed state during instruction execution

Status register (cont.)

Status registers state ST1	ST2	ST1 & ST2
00	00	No errors
01	00	The FDC cannot detect ID-Address Mark
01	01	The FDC can't find Data Address Mark or Deleted Data Address Mark
02	00	Write protected diskette
04	00	ID-field can't be read or sector is missing
04	01	In the diskettes ID-field cylinder is FFH, but not in command
04	10	In the diskette ID-field not equal to the one in command
10	00	Data transfer request not answered during correct time
20	00	CRC error in ID-field
20	20	CRC error in data field
80	00	An attempt to use a sector with too big number.
00	04	Scan condition is not fullfilled
00	08	Scan condition "equal" succeeded
00	40	Attempt to READ/COMPARE deleted data (other than READ DEL. DATA command)

Status register state	ST3
21H	Everything OK (drive=1)
41H	Not Ready, WP (drive=1)
61H	READY, WP (drive=1)
01H	Power is missing in the FDD

(x Bit-4 in ST3 can't be read, becaus of it's being active only after RECAL.

xx FAUL, Bit-7, not from the drive,but it is used to detect faults in FDC-card voltage)

2.8 INTERFACE/CONNECTORS:

A) Display Unit interface:
9 pin P D-connector

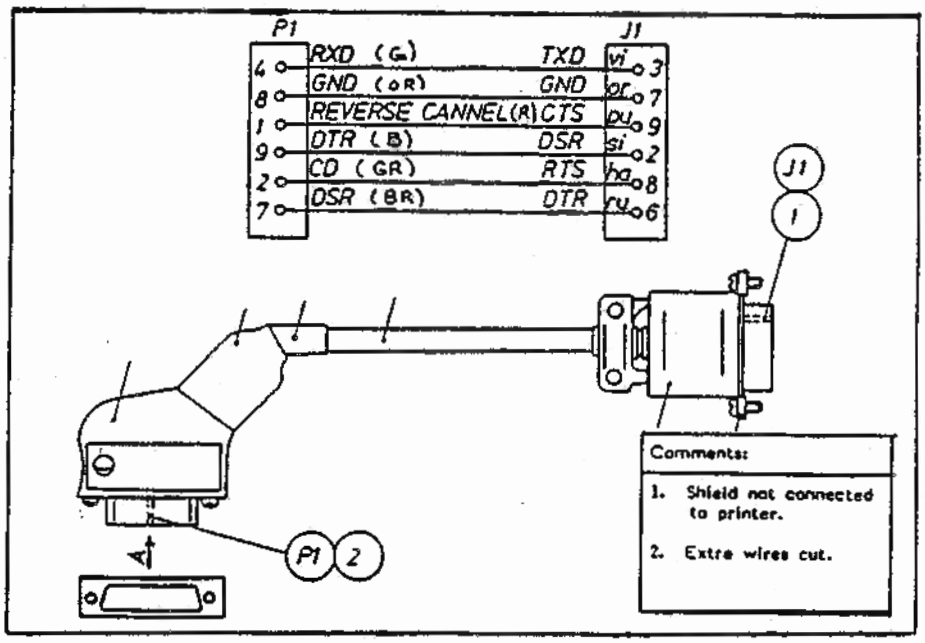
B) Keyboard interface:
15 pin P D-connector

C) Printer interface (or aux. interface)
CCITT V.24/V.28 (EIA RS232C)
asynchronous
75 - 9600 bit/s
9 pin S D-connector

PIN NUMBER	DIRECTION	SIGNAL	SIGNAL DESCRIPTION
1	IN	CTS	CLEAR TO SEND
2	OUT	RTS	REQUEST TO SEND
3	IN	RxD	RECEIVED DATA
4	OUT	TxD	TRANSMITTED DATA
7	OUT	DTR	DATA TERMINAL READY
8		GND	GROUND

NOTE: if CTS is not used, it need not be pulled up-

Printer Cable



D1) LINE INTERFACE (V.28)

CCITT V.24/V.28 (EIA R S232C)
Synchronous or asynchronous
75 - 9600 bit/s
25-pin P D-connector

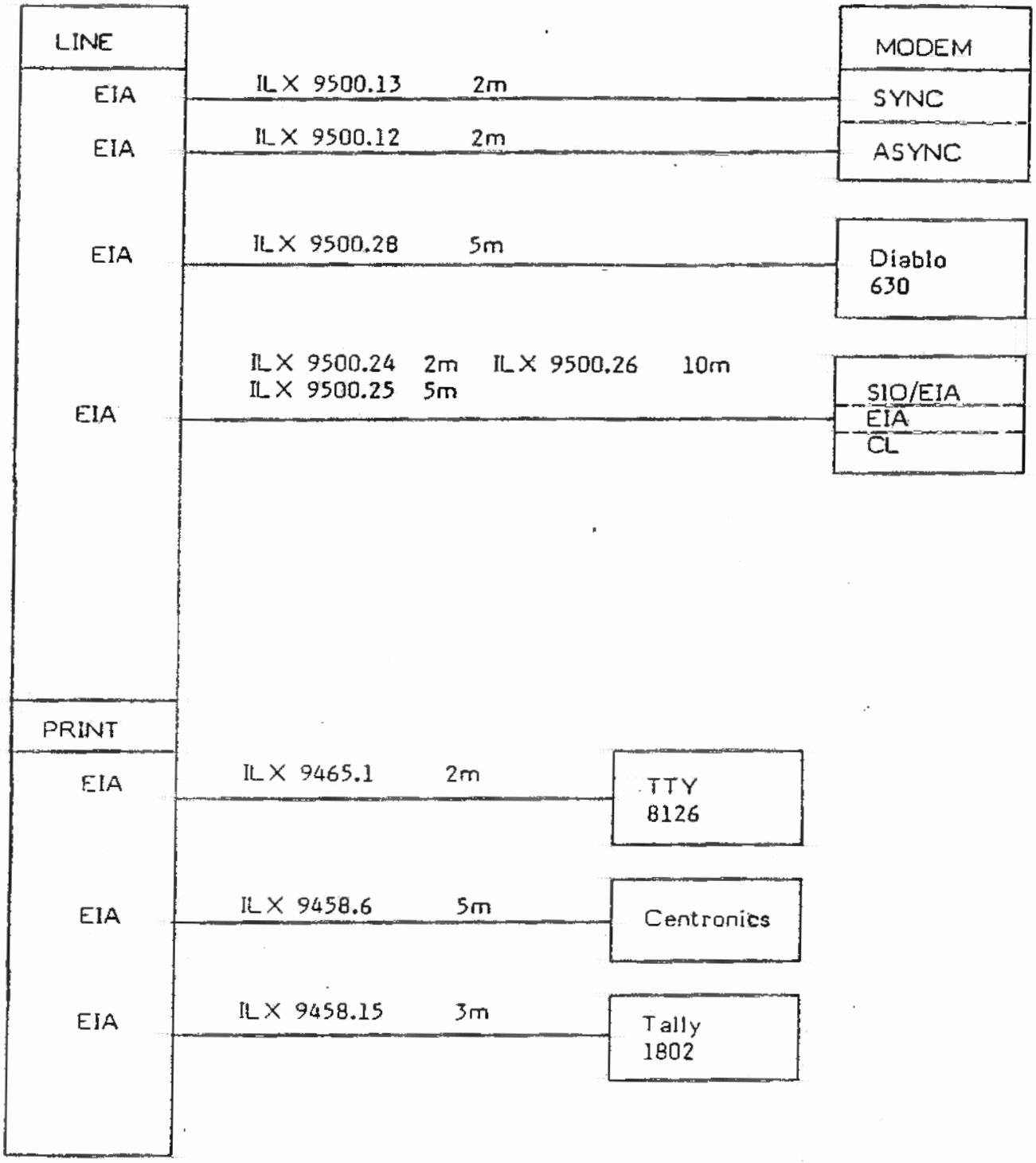
PIN NUMBER	DIRECTION	SIGNAL NAME	SIGNAL DESCRIPTION
2	OUT	TxD	TRANSMITTED DATA
3	IN	RxD	RECEIVED DATA
4	OUT	RTS	REQUEST TO SEND
5	IN *	CTS	CLEAR TO SEND
6	IN *	DSR	DATA SET READY
7		GND	GROUND
8	IN *	DCD	DATA CARRIER DETECTED
15	IN	TxC	TRANSMIT CLOCK
17	IN	RxC	RECEIVE CLOCK
20	OUT	DTR	DATA TERMINAL READY

*NOTE: If not used, must be pulled up, f.ex connected to DTR

D2) LINE INTERFACE (X.21)

Optional X.21 interface circuits are required to interface MikroMikko to Data Circuit-terminating Equipment (=DCE) for synchronous operation on Public Data Networks. This interface is connected to the same line connector as V.28.

PIN NUMBER	DIRECTION	SIGNAL	SIGNAL DESCRIPTION
9	IN	I (B)	(CD) Carrier Detect
10	IN	I (A)	(CD) Carrier Detect
11	IN	S (B)	(RxC and TxC) Receive
14	IN	S (A)	Clock and Transmit Clock
16	IN	R (B)	(RxD) Received Data
18	IN	R (A)	(RxD) Received Data
19	OUT	T (B)	(TxD) Transmitted Data
21	OUT	T (A)	(TxD) Transmitted Data
22	OUT	C (B)	(RTS) Request To Send
23	OUT	C (A)	(RTS) Request To Send



MIKROMIKKO, List of standard cable types

2.9 TEST & DIAGNOSTICS

Self-test

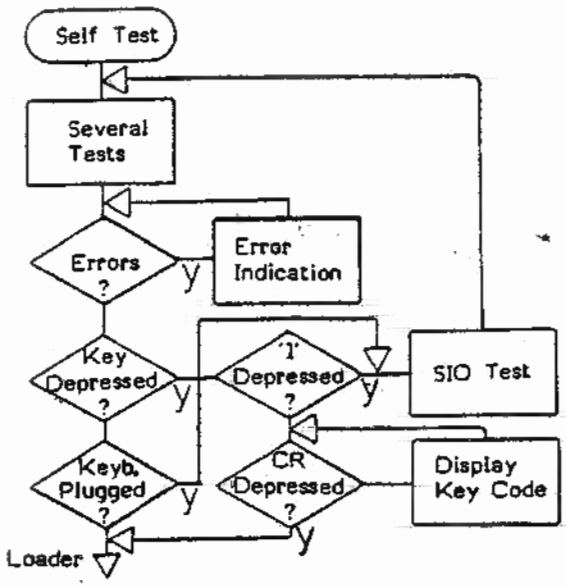
The power-on self-test locates on the PROM. It is executed every time power is switched on. It takes approc. 5 seconds. An error is indicated by a tone code and by an error message on the screen, if the system works so far. The power-up self-test can detect an indicate errors with a multi-part tone code with a minimum number of components which must be operating. This approach is called a minimum test kernel. Testing starts from the principal operations and continues to more sophisticated operations. When one area is detected to be in a satisfactory condition, the test kernel is expanded. This allows the use of more effective and complex testing an indication. The whole self-test is run by this method, logically continuing until all the system has been tested. Of course all the errors can not be detected by any test programs, but those which can be, are really easily detectable.

If no operation errors are found in the self-test, normal loading is carried out.

If a key is depressed immediately after power-on during the self-test execution (5 sec.) the key test is entered (PROM version can also be seen on the screen in this mode). Depressing the key generates corresponding key code on the screen. CR causes an exit to the loader program.

If key "1" is depressed during the self-test instead of any other key, SIO test is entered. This test requires the printer and line channel feed-back plugs presented in the chapter: Special Tools & Test Devices. If SIO works properly, the OK message appears, otherwise the information of an error status is displayed. Normal loading takes place after SIO test.

Continuous self-test can be achieved by plugging the keyboard connector with its feed back plug. This method can be used for system aging, with all three test plugs fixed. For detailed information of MicroMIKKO self-test see Micro-MIKKO Field Engineering Manual.



Self-test sequence

The self-test error codes are described in table

Test advancing →

Error Code (tone)	PROM	DMA Contr.	CRT Contr.	Display syncr.	RAM High	RAM High Refr.	RAM Low
Continuous tone	x	x	x	x	x	x	x
00	f	x	x	x	x	x	x
01	ok	f	f	x	f	x	x
02	ok	ok	f	x	f	x	x
03	ok	f	ok	x	f	x	x
04	ok	ok	ok	f	f	x	x
05	ok	f	f	x	ok	x	x
06	ok	ok	f	x	ok	x	x
07	ok	f	ok	x	ok	x	x
08	ok	ok	ok	f	ok	x	x
Display message & cont.tone	ok	ok	ok	ok	f	x	x
Display message & cont.tone	ok	ok	ok	ok	ok	x	f
Display message & cont.tone	ok	ok	ok	ok	ok	ok	f

x = not tested
f = failure
ok = test passed

error code:



start beep — short beeps (error code 03)

Self-test error codes

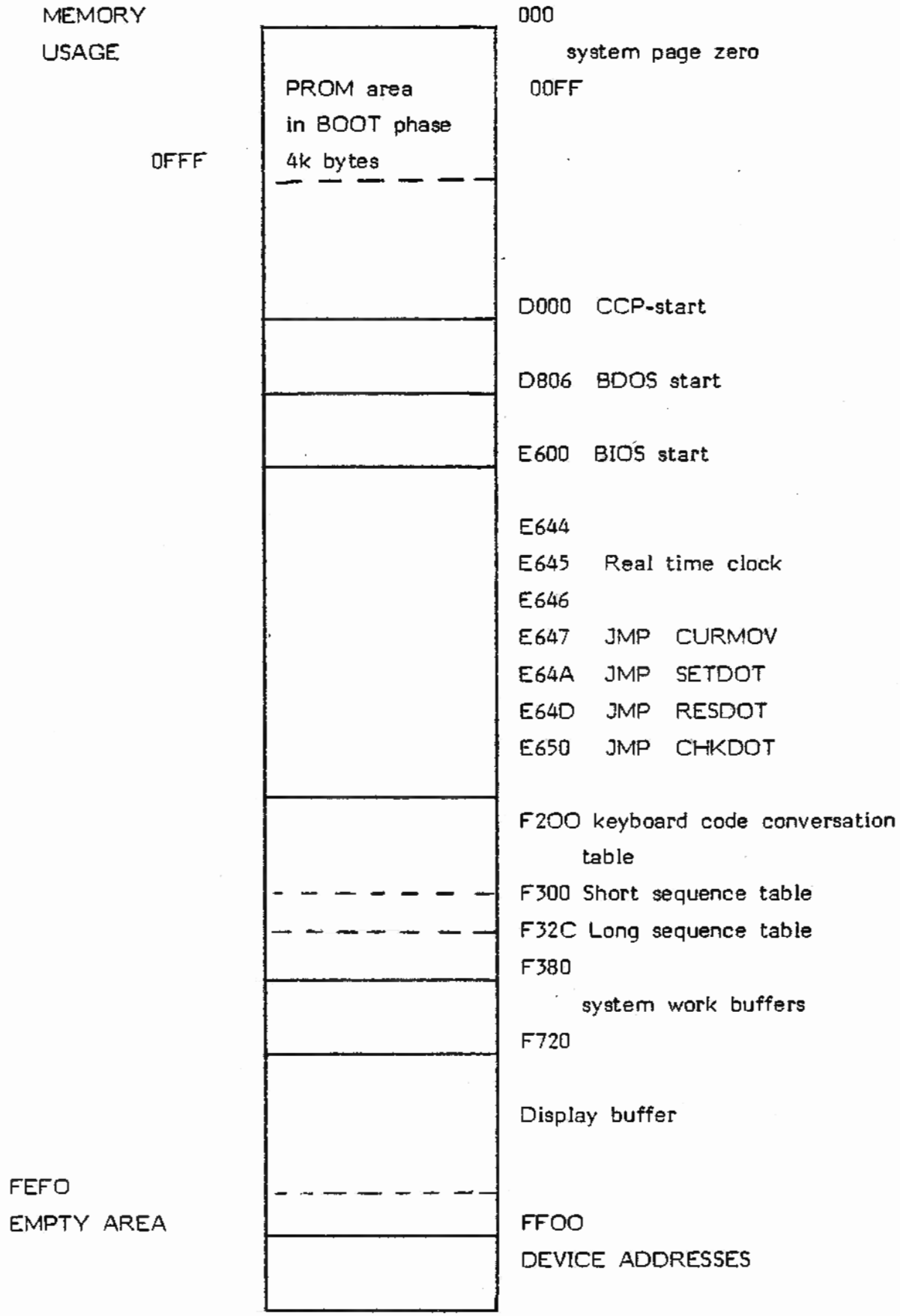
3.0 SOFTWARE

Operating system used in MicroMikko is CP/M rev. 2.2. CP/M is an abreviation of Control Program for Micrpcomputers. CP/M is a general purpose operating system for single user computers. It provides the facilities needed for program execution, file management and program development.

NOKIA CP/M is a customized version of the standard CP/M version 2.2 developed by DIGITAL RESEARCH. For detail information of CP/M facilities use CP/M manual set.

All information hereafter is based on CP/M version 2.2 Nokia customization number 2.2 if nothing else is said.

NOKIA CP/M 2.2 64K version



File Management

MicroMikko's operating system, CP/M, usually makes all needed operations to manage files. Files are accessed by names, consisting of 8 bytes, and 3 byte expansion. Number of filename entries is 64 for 160 kbyte drives and 128 for 320 kbyte drives. When the diskette is formatted, the filler character used is E5H .

In the 320 kbytes diskette two tracks are reserved for operating system and one track for directory information. A formatted 320 kbyte diskette has thus 308 kbytes free space for files. When the operating system is copied onto the diskette, still two kbytes are reserved for system extension file SYSTEXT. COM. This makes that the max. free space in a 320kbyte system diskette is 306 kbytes, and this again allows max. 2448 logical records (128 bytes each) to be written in a single file.

The corresponding values for 160 kbyte drives are:

Free space after formatting - 150 kbytes
Filespace in the system diskette - 149 kbytes
Max. number of logical records in a single
file in the system diskette - 1168 records

The smallest increment of a file is 2 kbytes in the 320 kbyte drive and 1 kbyte in 160 kbytes drive.

CP/M Bios parameters for 320 kbytes drive are set as follows:

Sectors per track	32
Block shift factor	4
Block mask	15
Extent mask	01
Disk size - 1	155
Directory max	127
Alloc 0 & 1	C000H
Check size	32
Track offset	02

Sector translate table

Physical sector	logical sector
1,4,7,2,	1,2,3,4,
5,8,3,6,	5,6,7,8,
(9,12,15,10,	(9,10,11,12,
13,16,11,14)	13,14,15,16)

Typical parameters for 320 kbyte drive

Step Rate Time	SRT = 10 ms
Head Unload Time	HUT = 320 ms
Head load Time	HLT = 36 ms

The diskette is formatted with following parameters

Recording mode	= MFM
Bytes per sector	N = 2
Sector per track	SC = 8
Length of GAP 3	GPL = 50H
Filler byte	D = E5H

Sector ID information consists of 4 bytes range

Cylinder number	C = 0 - 79
Headnumber	H = 0 or 1
Record number	R = 1 - 8
Bytes per sector	N = 2

Diskette Organization

The Nokia CP/M 2.2 system, version M4/2.1 is located on the system diskette as follows (addresses for standard 64k version):

Track 0
physical sector 1
    bytes 0-127: Cold Boot Loader
    bytes 128-511: CCP (D000H - D17FH)
phys. sector 2: CCP (D180H - D37FH)
phys. sector 3: CCP (D380H - D57FH)
phys. sector 4: CCP (D580H - D77FH)
phys. sector 5:
    bytes 0-127: CCP (D780H - D7FFH)
    bytes 128-511: BDOS (D800H - D97FH)
phys. sector 6: BDOS (D980H - DB7FH)
phys. sector 7: BDOS (DB80H - DD7FH)
phys. sector 8: BDOS (DD80H - DF7FH)

Track 1
physical sector 1: BDOS (DF80H - E17FH)
phys. sector 2: BDOS (E180H - E37FH)
phys. sector 3: BDOS (E380H - E57FH)
phys. sector 4:
    bytes 0-127: BDOS (E580H - E5FFH)
    bytes 128-511: BIOS (E600H - E77FH)
phys. sector 5: BIOS (E780H - E97FH)
phys. sector 6: BIOS (E980H - EB7FH)
phys. sector 7: BIOS (EB80H - ED7FH)
phys. sector 8: BIOS (ED80H - EF7FH)

Track 2 to 79 in file use;
File SYSTEXT. COM:

(8 logical sectors): BIOS (EF80H - F37FH)

The part of BIOS located after F380H is uninitialized buffer and work area and must not be read in from the diskette. The addresses F720H - FEEFH are used for video display buffer. FF00H - FFFFH is the device address area (memory-mapped I/O).

RESERVED LOCATIONS IN PAGE ZERO

Main memory page zero, between locations 00H and 0FFH, contains several segments of code and data which are used during CP/M processing. The code and data areas are given below for reference purposes.

Locations from to	Contents
0H - 0002H	Contains a jump instruction to the warm start entry point at location 4A03H+b. This allows a simple programmed restart (JMP 0000H).
3H - 0003H	Contains the Intel standard IOBYTE, which is included in the CBIOS.
4H - 0004H	Current default drive number (0=A,...,15=P).
5H - 0007H	Contains a jump instruction to the BDOS, and serves two purposes: JMP 0005H provides the primary entry point to the BDOS, as described in the manual "CP/M Interface Guide," and LHLD 0006H brings the address field of the instruction to the HL register pair. This value is the lowest address in memory used by CP/M (assuming the CCP is being overlayed). Note that the DDT program will change the address field to reflect the reduced memory size in debug mode.
8H - 000FH	not in use
10H - 12H	MPSC channel A interrupts
14H - 16H	MPSC Channel A interrupts

18H - 1AH MPSC channel A interrupts

1CH - 1EH MPSC channel A interrupts

1FH - 2BH NOT IN USE

2CH - 2EH FDC interrupt

30H C 9 = RET instruction. This address is jumped every time, when user program is reading one character from keyboard buffer. Character is in A.

31H - 33H NOT IN USE

34H - 36H Keyboard interrupt

38H - 003AH Restart 7 - Contains a jump instruction into the DDT or SID program when running in debug mode for programmed breakpoints, but not otherwise used by CP/M.

3CH - 3EH DMA interrupt

40H - 004FH 16 byte area reserved for scratch by CBIOS, but is not used for any purpose in the distribution version of CP/M

50H - 005BH (not currently used - reserved)

5CH - 007CH default file control block produced for a transient program by the Console Command Processor.

7DH - 007FH Optional default random record position

80H - 00FFH default 128 byte disk buffer (also filled with the command line when a transient is loaded under the CCP).

THE BIOS ENTRY POINTS

The entry points into the BIOS from the cold start loader and BDOS are detailed below. Entry to the BIOS is through a "jump vector" located as shown below. The jump vector is a sequence of 17 jump instructions which send program control to the individual BIOS subroutines.

The jump vector takes the form shown below, where the individual jump addresses are given to the left:

```
4A00H+b     JMP BOOT      ; ARRIVE HERE FROM COLD START LOAD
4A03H+b     JMP WBOOT     ; ARRIVE HERE FOR WARM START
4A06H+b     JMP CONST     ; CHECK FOR CONSOLE CHAR READY
4A09H+b     JMP CONIN     ; READ CONSOLE CHARACTER IN
4A0CH+b     JMP CONOUT    ; WRITE CONSOLE CHARACTER OUT
4A0FH+b     JMP LIST      ; WRITE LISTING CHARACTER OUT
4A12H+b     JMP PINCH     ; WRITE CHARACTER TO PUNCH DEVICE
4A15H+b     JMP READER    ; READ READER DEVICE
4A18H+b     JMP HOME      ; MOVE TO TRACK 00 ON SELECTED DISK
4A1BH+b     JMP SELDSK    ; SELECT DISK DRIVE
4A1EH+b     JMP SETTRK    ; SET TRACK NUMBER
4A21H+b     JMP SETSEC    ; SET SECTOR NUMBER
4A24H+b     JMP SETDMA    ; SET DMA ADDRESS
4A27H+b     JMP READ      ; READ SELECTED SECTOR
4A2AH+b     JMP WRITE     ; WRITE SELECTED SECTOR
4A2DH+b     JMP LISTST    ; RETURN LIST STATUS
4A30H+b     JMP SECTRAN   ; SECTOR TRANSLATE SUBROUTINE
```

Each jump address corresponds to a particular subroutine which performs the specific function, as outlined below. There are three major divisions in the jump table: the system (re)initialization which results from calls on BOOT and WBOOT, simple character I/O performed by calls on CONST, CONIN, CONOUT, LIST, PUNCH, READER, and LISTST, and diskette I/O performed by calls on HOME, SELDSK, SETTRK, SETSEC, SETDMA, READ, WRITE, and SECTRAN.

All simple character I/O operations are assumed to be performed in ASCII, upper and lower case, with high order (parity bit) set to zero. An end-of-file condition for an input device is given by an ASCII control-z (1AH). Peripheral device are seen by CP/M as "logical" devices, and are assigned to physical devices within the BIOS:

The characteristics of each device are

CONSOLE — The principal interactive console which communicates with the operator, accessed through CONST, CONIN, and CONOUT. Typically, the CONSOLE is a device such as a CRT or Teletype.

LIST — The principal listing device, if it exists on your system, which is usually a hard-copy device, such as a printer or Teletype.

PUNCH — The principal tape punching device, if it exists, which is normally a high-speed paper tape punch or Teletype.

READER — The principal tape reading device, such as a simple optical reader or Teletype.

Note that a single peripheral can be assigned as the LIST, PUNCH, and READER device simultaneously.

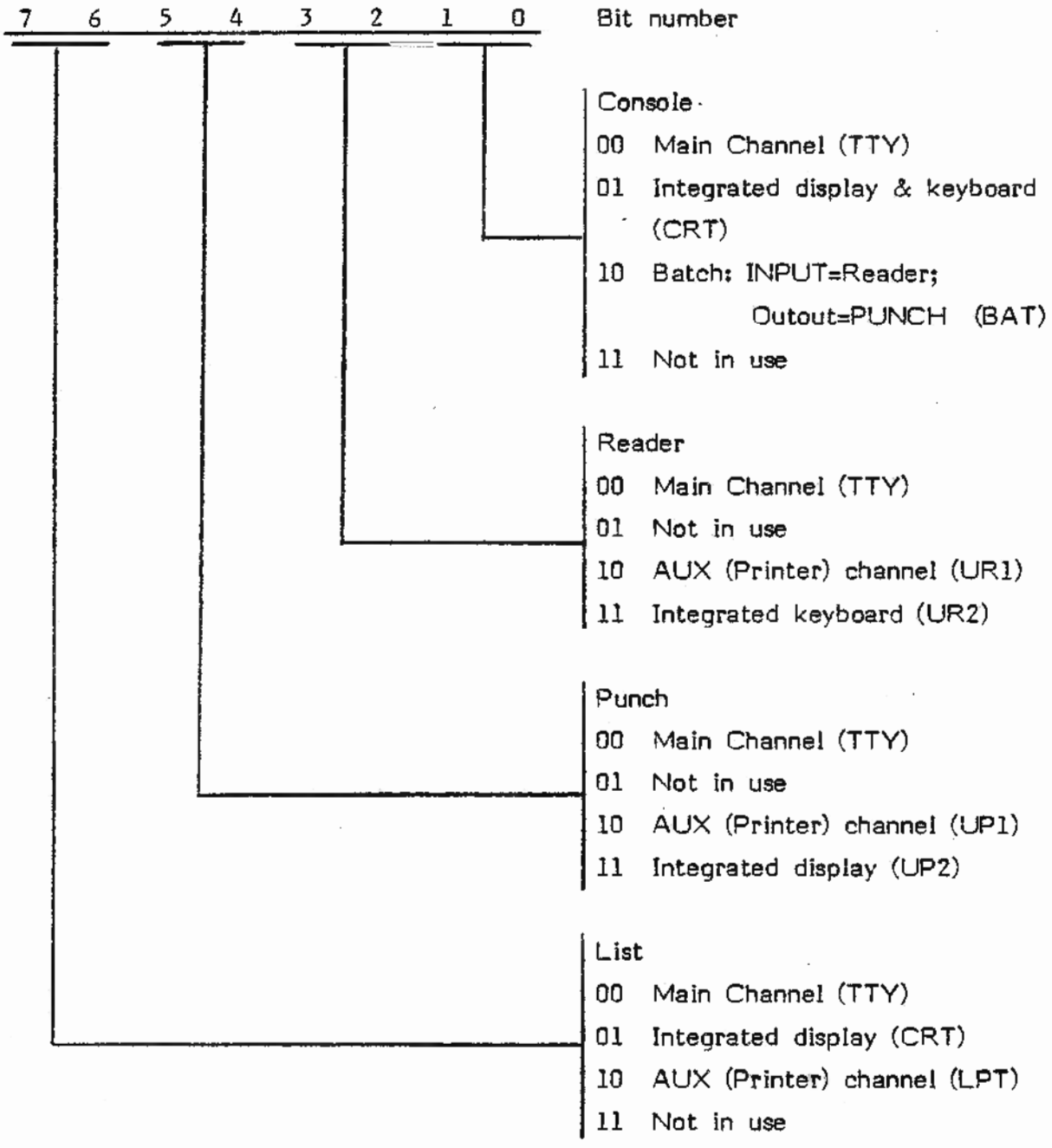
The IOBYTE function creates a mapping of logical to physical devices which can be altered during CP/M processing (see the STAT command). The definition of the IOBYTE function corresponds to the Intel Standard as follows: a single location in memory (currently location 003H) is maintained, called IOBYTE, which defines the logical to physical device mapping which is in effect at a particular time. The mapping is performed by splitting the IOBYTE into four distinct fields of two bits each, called the CONSOLE, READER, PUNCH, and LIST fields, as shown below.

	most significant			least significant
IOBYTE AT 0003H	LIST	PUNCH	READER	CONSOLE
	bits 6,7	bits 4,5	bits 2,3	bits 0,1

The value in each field can be in range 0-3, defining the assigned source or destination of each logical device. The values which can be assigned to each field are given below

IOBYTE

Default value of IOBYTE is 81 H



SYSTEM FUNCTION SUMMARY.

FUNC	FUNCTION NAME	INPUT PARAMETERS	OUTPUT RESULTS
0	System Reset	none	none
1	Console Input	none	A = char
2	Console Output	E = char	none
3	Reader Input	none	A = char
4	Punch Output	E = char	none
5	List Output	E = char	none
6	Direct Console I/O	see def	see def
7	Get I/O Byte	none	A = IOBYTE
8	Set I/O Byte	E = IOBYTE	none
9	Print String	DE = .Buffer	none
10	Read Console Buffer	DE = .Buffer	see def
11	Get Console Status	none	A = 00/FF
12	Return Version Number	none	HL= Version*
13	Reset Disk System	none	see def
14	Select Disk	E = Disk Number	see def
15	Open File	DE = .FCB	A = Dir Code
16	Close File	DE = .FCB	A = Dir Code
17	Search for First	DE = .FCB	A = Dir Code
18	Search for Next	none	A = Dir Code
19	Delete File	DE = .FCB	A = Dir Code
20	Read Sequential	DE = .FCB	A = Err Code
21	Write Sequential	DE = .FCB	A = Err Code
22	Make File	DE = .FCB	A = Dir Code
23	Rename File	DE = .FCB	A = Dir Code
24	Return Login Vector	none	HL= Login Vect*
25	Return Current Disk	none	A = Cur Disk#
26	Set DMA Address	DE = .DMA	none
27	Get Addr(Alloc)	none	HL= .Alloc
28	Write Protect Disk	none	see def
29	Get R/O Vector	none	HL= R/O Vect*
30	Set File Attributes	DE = .FCB	see def
31	Get Addr(disk parms)	none	HL= .DPB
32	Set/Get User Code	see def	see def
33	Read Random	DE = .FCB	A = Err Code
34	Write Random	DE = .FCB	A = Err Code
35	Compute File Size	DE = .FCB	r0, r1, r2
36	Set Random Record	DE = .FCB	r0, r1, r2

LINE AND PRINTER CHANNEL FREQUENCY CHANGE

The transmission and reception speeds of the two communications lines can be changed. The original values are 9600 bauds. The main line has separate clock generators for transmitter and receiver, while in the printer channel the speeds must be same in both directions.

The clock generator register addresses are:

FF30H	Main Line/Transmitter
FF31H	Main Line/Receiver
FF32H	Printer Line

To change the clock speed from the default 9600 bd, one must write two consecutive bytes to one of the addresses given above. No reading is legal, thus the "S" command in DDT cannot be used.

The speed defining word for normal asynchronous connections can be calculated with the formula

$$\text{line speed} = \frac{96000\text{bd}}{\text{DDEE}}$$

where DDEE is the absolute value for the speed word. After calculating the value, convert it to hexadecimal and write it to the given clock address LOW-ORDER BYTE FIRST. If the second byte should be zero, it must anyhow be written.

An example of a BASIC program to set a new transmitter frequency:

```
 5  REM SET THE FREQUENCY TO 1200 BAUDS
10  POKE &HFF30, &H50
15  POKE &HFF30, &H00
20  END
```

An 8085 Assembly language program would look like this:

```
DDEE  EQU  50H
--
LXI   H, 0FF30H
MVI   M. LOW DDEE
MVI   M, HIGH DDEE
```

Any changed line frequency will remain valid until next cold boot or power off.

REAL-TIME CLOCK

Nokia CP/M 2.2 supplies the user with a crystal-driven clock. The clock can be used through reading (and writing) a 24-bit (three bytes) binary counter.

The Clock counter is located in three consecutive bytes starting from BIOS+44H (E644H in M2/2.1 and M4/2.1). The bytes come in ascending order, i.e. the least significant byte is First.

The system increments the Clock bytes every 20 milliseconds. This is the only connection: the system itself does not utilize the Clock and thus the user may also give the counter new values when necessary.

When not reset, the 2nd byte of the clock is incremented every 5.12 seconds and the 3rd byte about every 22 minutes. The whole counter is reset after about 4 days.

The Clock bytes can be read or written at any time. Notice, however, that the counter may be incremented and a carry to a higher-order byte generated between two byte read commands. The hazard can be avoided through reading the clock value twice (must be same) or using "interrupts disabled" state in an assembly language level program.

KEYBOARD CODE CONVERSION

When the user depresses a key on the keyboard, it sends an eight-bit code to the BIOS. This physical code is used as an index to look up the associated logical code, also a byte, in a special Conversion Table. The Conversion Table is located at BIOS+0D00H (=F200H) and the physical code is used directly as a byte index. All codes from 0 to FFH are in this phase equivalent.

The action to be taken after the conversion depends on the logical code produced:

Note: This table is valid from version M*/2.2

Logical Code	Action
0-60H	Insert the byte to keyboard buffer
61H-70H	(Small letters) Insert to KB buffer (after conversion to a capital letter, if CAPS LOCK is on)
7EH-7FH	Insert to KB buffer
80H-89H	Find the long sequence function 1-10, insert the found string o KB buffer
8AH-9FH	Find the short sequence function 11-32, insert the found byte(s) to KB buffer
A0H-C9H	Insert to KB buffer (non ASCII)
CAH	Quit any function being executed and jump to cold boot program. This operation is not inhibited by keyboard lock function.
CBH-E3H	Insert to KB buffer (non ASCII)
E4H	Set/Reset CAPS LOCK function (no code generated to KB buffer)

E5H-FEH Insert to KB buffer (non ASCII)

FFH No operation (dummy key)

The physical keycodes can be found in chapter 2.3. With standard MicroMikko keyboard , the logical codes generated by the alphanumeric keys are the same as the physical ones. The numeric field codes are converted to ASCII numbers and decimal point. The function keys F1 and F6-F10 produce the long function sequences . The five cursor keys, functionkeys F2 - F5, and all the upmost row with SHIFT use the short ones .

Any changes made directly to the conversion table remain valid until next cold boot or power-off. The conversion table can be permanently modified by altering the file SYSTEXT.COM; the relative address of the table from the start of this file is 280H.

Long Sequence Functions

There are 10 user-selectable keys that produce an ASCII string to the keyboard buffer at a single keystroke. The individual lengths of these sequences are arbitrary; in total there must be no more than 74 characters (+10 separators).

The function sequences are located one after another, separated by single hexadecimal FF bytes. The sequence table starts at BIOS+E2CH (=F32CH) and may extend up to BIOS+E7FH (=F37FH). In this area there must always exist at least ten separate bytes (FFH) to avoid system overflow.

When the BIOS program detects the logical code (see chapter KB Code Conversion) 80H, it inserts to the keyboard buffer all bytes found in the table before the first separator. Code 81H causes the string between first and second separator to be inserted tec., up to logical code 89H. Any sequence may also be empty.

The bytes found in the function table are brought as such into the keyboard buffer. No checks for control or non-ASCII characters are made.

The Long Sequence Function Table starts at relative address 3ACH in the file SYSTEXT.COM where any permanent changes should be made.

Short Sequence Functions

The user can define 22 keys to produce a sequence of 0 to 2 characters each. This is sufficient for most control sequencies; in addition, non-ASCII characters can be sent to the user's program.

The short functions are activated by logical key codes 8AH to 9FH. Each code reserves two consecutive bytes in the function table starting at BIOS+E00H (=F300H). The two bytes corresponding to the found logical code are inserted as such into the keyboard buffer, unless:

1) the first byte is FFH: no byte is inserted (the second byte is omitted)
2) the second byte is FFH: only the first byte is inserted into the KB buffer.

Because no checks for the codes found in the table are made, the short sequence functions can be used to override CAPS LOCK function.

The Short Sequence Function Table starts at relative address 380H in the file SYSTEXT.COM.

EXAMPLE

Making Permanent Changes to Keyboard Conversion & Function Tables

Remove the cold boot Function from the keyboard, system to be changed in drive B

```
A>STAT B:SYSTEXT.COM  $R/W
A>DDT B:SYSTEXT.COM
DDT VERS 2.2
NEXT PC
0500  0100
-S44A
044A CA FF
044B 8F.
- C
A>SAVE 4 SYSTEXT.COM
```

```
A>STAT SYSTEXT.COM $SYS
A>STAT SYSTEXT.COM $R/O
A>
```

MENU

The Menu program is supplied in source code as well as in executable .COM form. The standard version is only skeletal and can be augmented by a skilled user.

The operation of Menu is based on patching the command to be executed into the CCP area and jumping to CCP. This means that executing a similar routine from BASIC interpreter is not possible, because the CCP area is used for other purposes.

To add in new functions to the menu, you have to change the limit checks ("CPI 2", after "SUI '1'"), add the new entries to the address table "ADDS", and form the entries in the form

<the command to be given, in ASCII>,
<00 (a separator instead of carriage return)>,
<executable 8085 assembly code: any subprogram to be executed before entering the selected program and a jump to one of the entries ADISK or BDISK>

To change the starting message to be displayed, add or change lines in the string starting at "STMESS". Notice that the "$" ends the message.

Menu or any other program can be activated automatically at cold or warm boot through making the patch mentioned above permanently to the system residing on the diskette. You can use Inittask to make this update. Inittask is a program which is delivered with system diskette, and can be started with command INITTASK cr.

```
                        -- A SKELETAL MENU PROGRAM --
                        DESIGNER: ARI RYTKOLA 1982-03-29
                        >> ADDRESSES ARE VALID FOR VERSION MX/2.1 ONLY!

0005 =                  BDOS    EQU     0005H   ;BDOS ENTRY POINT

0004 =                  CDISK   EQU     0004H   ;CURRENT DISK SELECTION
D000 =                  CCP     EQU     0D000H  ;CCP BEGINNING ADDRESS
D007 =                  COUNT   EQU     CCP+7   ;START MESSAGE LENGTH
D008 =                  MESSA   EQU     CCP+8   ;MESSAGE AREA START
D088 =                  CURRENT EQU     CCP+88H ;CURRENT POINTER TO MESS BUFFER

E600 =                  COLD    EQU     0E600H  ;COLD START ENTRY POINT


0001 =                  CONIN   EQU     1       ;CONSOLE INPUT
0009 =                  PRINT   EQU     9

001A =                  CLR     EQU     1AH     ;CLEAR SCREEN
000D =                  CR      EQU     0DH     ;CARRIAGE RETURN
000A =                  LF      EQU     0AH     ;LINE FEED
0009 =                  TAB     EQU     09H     ;TABULATOR


0100 =                  TPA     EQU     100H

0100                    ORG     TPA
0100 31CF01     START:  LXI     SP,STACK        ;USE LOCAL STACK
0103 113901             LXI     D,STMESS        ;PRINT START MESSAGE
0106 0E09               MVI     C,PRINT
0108 CD0500             CALL    BDOS

010B 0E01               MVI     C,CONIN         ;WAIT FOR KEYBOARD RESPONSE
010D CD0500             CALL    BDOS
0110 D631               SUI     '1'             ;NORMALIZE
0112 FE02               CPI     2
0114 D20001             JNC     START           ;WRONG ANSWER: LOOP BACK

0117 4F                 MOV     C,A             ;CALCULATE BRANCH ADDRESS
0118 0600               MVI     B,0
011A 218D01             LXI     H,ADDS
011D 0909               DAD B!  DAD B
011F 5E2356             MOV E,M! INX H! MOV D,M

0122 06FF       FILL:   MVI     B,0FFH          ;READ THE COMMAND TO CCP BUFFER
0124 2188D0             LXI     H,MESSA
0127 2288D0             SHLD    CURRENT         ;UPDATE 'CURRENT POINTER'
012A EB                 XCHG

012B 7E12       LOOP:   MOV A,M! STAX D         ;MOVE BYTE BY BYTE
012D 231304             INX H! INX D! INR B
0130 B7C22B01           ORA A!  JNZ LOOP

0134 7808207D0          MOV A,B! STA COUNT      ;STORE MESSAGE LENGTH

0138 E9                 PCHL                    ;BRANCH TO PROGRAM AFTER COMMAND


0139 1A0A       STMESS: DB      CLR,LF          ;START MESSAGE
013B 094D204920         DB      TAB,'M I C R O M I K K O   M E N U'
0159 0D0A0A0A           DB      CR,LF,LF,LF
015D 09313A2042         DB      TAB,'1: BASIC',CR,LF,LF
0169 09323A2043         DB      TAB,'2: COMMAND LEVEL ',CR,LF,LF
017E 0953454C45         DB      TAB,'SELECT (1-2)?'
018C 24                 DB      '$'

018D 91019B01   ADDS:   DW      C1,C2


0191 4D42415349C1:      DB      'MBASIC',0
0198 C39F01             JMP     ADISK

019B 00         C2:     DB      0
019C C39F01             JMP     ADISK


019F 3E00       ADISK:  MVI     A,0             ;SELECT DRIVE A AS DEFAULT
01A1 320400             STA     CDISK
01A4 4F                 MOV     C,A
01A5 C300D0             JMP     CCP

01A8 3E01       BDISK:  MVI     A,1             ;DRIVE B
01AA 320400             STA     CDISK
01AD 4F                 MOV     C,A
01AE C300D0             JMP     CCP

01B1                    DS      30
                STACK:
01CF                    END
```

TRANSMIT & RECEIVE

These are programs written with assembler and delivered with system diskette.

Programs can be used to move data from one MicroMikko to another with help of Main Channel and standard EIA cable. Program source code is also included with system diskette, thus allowing changes to the program.

The main idea of delivering a such program with systemdiskette, is to give the users of MicroMikko the possibility to make similar type of programs for other types of computers, and this way make data transfers possible.

NOTE: These programs modify MPSC preset values to eight databits, and this definition is valid up to next modification (such as COLD-BOOT).

```
                ; R E C E I V E   F I L E   F R O M   T H E   I / O - L I N E
                ; ***********************************************************

                ; COMMAND "RECEIVE X.Y" RECEIVES A FILE FROM I/O-LINE INTO
                ; MEMORY AND AFTER THAT SAVES IT WITH NAME X.Y .
                ;
                ; THE FIRST TWO BYTES IN THE COMING DATA ARE SECTOR COUNT
                ; BYTES AND THE LAST BYTE IS THE PARITY BYTE,WHICH IS
                ; ALSO CHECKED.

                ; THIS PROGRAM IS USED IN MICROMIKKO COMPUTERS
                ;       LAST EDITION 1982-03-24 BY AR

                ;TITLE "RECEIVE"


0000 =          WBOOT   EQU     0000H   ;SYSTEM REBOOT
0005 =          BDOS    EQU     0005H   ;BDOS ENTRY POINT
005C =          FCB1    EQU     005CH   ;FILE CONTROL BLOCK
0080 =          DBUFF   EQU     0080H   ;DEFAULT BUFFER
0100 =          TPA     EQU     0100H   ;BEGINNING OF TPA

0001 =          CONIN   EQU     01      ;CONSOLE INPUT
0003 =          READER  EQU     03      ;READER INPUT
0004 =          PUNCH   EQU     04      ;PUNCH OUTPUT
0007 =          GETIO   EQU     07      ;GET I/O -BYTE
0008 =          SETIO   EQU     08      ;SET I/O -BYTE
0009 =          PRINTF  EQU     09      ;PRINT BUFFER FUNC#
000B =          CONST   EQU     11      ;GET CONSOLE STATUS
0010 =          CLOSEF  EQU     16      ;CLOSE FILE FUNC#
0013 =          DELETEF EQU     19      ;DELETE FILE FUNC#
0015 =          WRITEF  EQU     21      ;SEQUENTIAL WRITE
0016 =          MAKEF   EQU     22      ;MAKE FILE FUNC#
001A =          SDMAF   EQU     26      ;SET DMA ADDRESS
FF12 =          SIOAC   EQU     0FF12H  ;MPSC CH A CONTROL/STATUS
FF31 =          TMRE    EQU     0FF31H  ;TIMER CHANNEL FOR RECEIVE
0500 =          BEG     EQU     0500H   ;BEGIN OF BUFFER AREA
00FF =          MASK    EQU     0FFH            ;MASK FOR MAINLINE
FFE2 =          AMASK   EQU     0FFE2H          ;MASK BYTE ADDRESS FOR MAIN LINE



0100                    ORG     TPA             ;BEGINNING OF TPA
0100 210000             LXI     H,0
0103 39                 DAD     SP
0104 22A502             SHLD    OLDSP           ;SAVE OLD STACK POINTER
0107 310002             LXI     SP,STACK        ;LOCAL STACK

                ;       FIRST UPDATE THE MPSC FOR 8 DATA BITS ,SET THE TIMER CHANNEL
                ;       TO CORRECT BAUD RATE AND CHECK THE I/O -BYTE DEFINITION

010A F3                 DI                      ;UPDATE THE MULTI-PROTOCOL CIRCUIT
010B 3A12FF             LDA     SIOAC
010E 3E03               MVI     A,03H           ;TO RECEIVE 8 DATA BITS
0110 3212FF             STA     SIOAC
0113 3EC1               MVI     A,0C1H
0115 3212FF             STA     SIOAC

0118 2131FF             LXI     H,TMRE          ;SET THE BAUD RATE FOR RECEIVING
011B 360A               MVI     M,0AH           ;SET FIRST LSB OF THE CODE
011D 3600               MVI     M,00H           ;AND THEN MSB (000A4=9600 BAUD)

011F FB                 EI

0120 0E07               MVI     C,GETIO         ;GET I/O -BYTE
0122 CD0500             CALL    BDOS

0125 32A302             STA     IOBYTE          ;SAVE

0128 3E00               MVI     A,00000000B     ;SET CONSOLE TO MAIN LINE
012A 5F                 MOV     E,A
012B 0E08               MVI     C,SETIO         ;SET I/O -BYTE
012D CD0500             CALL    BDOS

0130 0E0B       EMLOOP: MVI     C,CONST         ;GET CONSOLE STATUS
0132 CD0500             CALL    BDOS
0135 B7                 ORA     A
0136 CA4101             JZ      NEWIO           ;WHEN ZERO, THE BUFFER IS EMPTY

0139 0E01               MVI     C,CONIN
013B CD0500             CALL    BDOS            ;READ ONE CHARACTER
013E C33001             JMP     EMLOOP          ;REPEAT

0141 3AA302     NEWIO:  LDA     IOBYTE          ;THE BUFFER IS EMPTY: RESET CONSOLE
```

```
0144 E6F3               ANI     11110011B       ;STATUS TO (CRT)
0146 5F                 MOV     E,A
0147 0E09               MVI     C,SETIO
0149 CD0500             CALL    BDOS

014C 3AF2F1             LDA     AMASK           ;READ THE MASK OF THE MAIN I/O -LINE
014F 32A402             STA     OMASK           ;SAVE
0152 3EFF               MVI     A,MASK          ;SET THE MAIN I/O-LINE TO 8 DATA BITS
0154 32F2F1             STA     AMASK


0157 AF                 XRA     A
0158 32A702             STA     COUNT           ;RESET 128 -COUNTER
015B 32AC02             STA     PARITY          ;AND PARITY -COUNTER

                ;
                ;       REMOVE DESTINATION FILE IF PRESENT AND RECEIVE
                ;       THE SECTOR COUNT, BYTES, PARITY COUNTING IS STARTED
                ;
015E 11C000     CONTIN  LXI     D,FCBL          ;DESTINATION FILE
0161 0E13               MVI     C,DELETEF       ;REMOVE IF PRESENT
0163 CD0500             CALL    BDOS
0166 11C000             LXI     D,FCBL
0169 0E16               MVI     C,MAKEF         ;CREATE THE FILE
016B CD0500             CALL    BDOS
016E 11A002             LXI     D,NODIR         ;NO SPACE -MESSAGE
0171 3C                 INR     A               ;255 BECOMES 0
0172 CC1A03             CZ      FINIS           ;DONE IF NO DIR SPACE

0175 CD4502             CALL    SIOTES          ;TAKE A CHAR FROM SIO
0178 32A802             STA     COUN            ;SAVE SECTOR COUNT LOW  ORDER BYTE
017B 32AA02             STA     SAVECO
017E 47                 MOV     B,A
017F 3AAC02             LDA     PARITY          ;COUNT PARITY
0182 A8                 XRA     B
0183 32AC02             STA     PARITY
0186 CD4502             CALL    SIOTES
0189 32A902             STA     COUN+1          ;HIGH ORDER BYTE
018C 32AB02             STA     SAVECO+1
018F 47                 MOV     B,A
0190 3AAC02             LDA     PARITY          ;PARITY COUNTING
0193 A8                 XRA     B
0194 32AC02             STA     PARITY



                ;
                ;       RECEIVE THE DATA INTO MEMORY
                ;
0197 110005     COPY    LXI     D,BEG           ;START ADDRESS TO DE

019A CD4502     CON     CALL    SIOTES          ;TAKE A BYTE FROM SIO
019D 12                 STAX    D               ;SAVE
019E 47                 MOV     B,A
019F 3AAC02             LDA     PARITY          ;LOAD PARITY COUNTER
01A2 A8                 XRA     B               ;AND COUNT NEW PARITY VALUE
01A3 32AC02             STA     PARITY
01A6 13                 INX     D

01A7 3AA702             LDA     COUNT           ;LOAD BYTE COUNTER
01AA 3C                 INR     A
01AB 32A702             STA     COUNT
01AE FE80               CPI     128             ;ONE SECTOR HAS COME ?
01B0 C29A01             JNZ     CON
01B3 AF                 XRA     A
01B4 32A702             STA     COUNT
01B7 CD3302             CALL    VERIF
01BA D29A01             JNC     CON             ;CARRY=1 MEANS: ALL RECEIVED

                ;
                ;       THE PARITY VERIFICATION
                ;
01BD CD4502     SAVE    CALL    SIOTES
01C0 47                 MOV     B,A             ;GET THE PARITY BYTE SENDED
01C1 11B402             LXI     D,PERROR        ;PARITY ERROR -MESSAGE
01C4 3AAC02             LDA     PARITY
01C7 B8                 CMP     B               ;COMPARE PARITY BYTES
01C8 C41A03             CNZ     FINIS           ;IF NOT SAME

                ;
                ;       WRITE THE FILE FROM MEMORY
                ;
01CB 210005     STOPPI  LXI     H,BEG           ;START ADDRESS
01CE 22A502             SHLD    DMAD
01D1 3AAA02             LDA     SAVECO          ;SECTOR COUNT LSB
01D4 32A802             STA     COUN
01D7 3AAB02             LDA     SAVECO+1        ;SECTOR COUNT MSB
01DA 32A902             STA     COUN+1
01DD EB         WRI     XCHG                    ;ADDRESS TO DE
01DE 0E1A               MVI     C,SDMAF         ;SET DMA ADDRESS
01E0 CD0500             CALL    BDOS
```

```
01E3 115C00             LXI     D,FCBL          ;DESTINATION FILE CONTR. BLOCK
01E6 0E15               MVI     C,WRITEF        ;WRITE RECORD
01E8 CD0500             CALL    BDOS
01EB 1140D2             LXI     D,SPACE         ;WRITE ERROR -MESSAGE
01EE B7                 ORA     A               ;00 IF WRITE OK
01EF C41402             CNZ     FINIS           ;END IF SO
01F2 2AA502             LHLD    DMAD            ;CURRENT ADDRESS
01F5 118000             LXI     D,0080H         ;<HL>:=<HL>+80H
01F8 19                 DAD     D
01F9 22A502             SHLD    DMAD            ;SAVE INCREMENTED ADDRESS
01FC CD3302             CALL    VERIF           ;IS ALL SAVED ?
01FF DA0302             JC      WRE             ;CARRY=1 MEANS YES
0202 2AA502             LHLD    DMAD            ;GET ADDRESS BACK TO HL
0205 C30001             JMP     WRI

0208 115C00     WRE     LXI     D,FCBL          ;DESTINATION FILE...
020B 0E10               MVI     C,CLOSEF        ;CLOSE DESTINATION FILE
020D CD0500             CALL    BDOS
0210 1172D2             LXI     D,WRPROT        ;CLOSE ERROR -MESSAGE
0213 3C                 INR     A               ;255 BECOMES 00
0214 CC1402             CZ      FINIS           ;SHOULDN'T HAPPEN

0217 11E1D2             LXI     D,NORMAL        ;NORMAL END OF THE PROGRAM
021A 0E09       FINIS   MVI     C,PRINTF
021C CD0500             CALL    BDOS            ;WRITE MESSAGE

021F 3AA302             LDA     IOBYTE          ;GET THE OLD IOBYTE DEFINITION
0222 5F                 MOV     E,A
0223 0E08               MVI     C,SETIO         ;SET I/O -BYTE
0225 CD0500             CALL    BDOS

0228 3AA402             LDA     OMASK
022B 32F2F1             STA     AMASK           ;SET OLD MASK FOR MAIN LINE

022E 2AAE02             LHLD    OLDSP           ;LOAD OLD STACK
0231 F9                 SPHL
0232 C9                 RET

                ;
                ;       DECREMENT AND VERIFY COUNTER
                ;
0233 E5         VERIF   PUSH    H
0234 2AA802             LHLD    COUN            ;LOAD COUNTER


0237 2B                 DCX     H               ;AND DECREMENT
0238 22A802             SHLD    COUN
023B 7C                 MOV     A,H
023C B5                 ORA     L               ;IS COUNTER ZERO ?
023D E1                 POP     H
023E CA4302             JZ      ZERO
0241 AF                 XRA     A               ;RESET CARRY
0242 C9                 RET
0243 37         ZERO    STC                     ;SET CARRY TO MEAN:
0244 C9                 RET                     ;COUNTERS = 0

                ;       SIOTES RECEIVES A BYTE FROM LINE VIA BDOS CALL

0245 D5         SIOTES: PUSH    D
0246 0E03               MVI     C,03H           ;READ A BYTE FROM READER(=MAIN LINE)
0248 CD0500             CALL    BDOS
024B D1                 POP     D
024C C9                 RET

                ;       CONSOLE MESSAGES
024D 4E4F20444952NODIR  DB      'NO DIRECTORY SPACES$'
0260 4F55542044FSPACE   DB      'OUT OF DATA SPACES$'
0272 575249544557RPROT  DB      'WRITE PROTECTED ?$'
0284 504152495459ERROR  DB      'PARITY ERRORS$'
0291 5452414E53NORMAL   DB      'TRANSFER COMPLETE$'
                ;       DATA AREAS

0243            IOBYTE: DS      1               ;I/O -BYTE
0244            OMASK:  DS      1               ;MASK FOR MAIN LINE
0245            DMAD    DS      2               ;DMA ADDRESS IN WRITE
0247            COUNT   DS      1               ;128 -COUNTER
0248            COUN    DS      2               ;SECTOR COUNTER
024A            SAVECO  DS      2               ;SAVED SECTOR COUNTER
024C            PARITY  DS      1               ;PARITY COUNTER
024D            COUNT   DS      1               ;SECTOR COUNTER
024E            OLDSP   DS      2               ;OLD STACK POINTER
                ;
0280            STACK:  DS      32              ;16 LEVEL STACK
0207                    END
```

```
                        ; T R A N S M I T   F I L E   T O   T H E   I / O - L I N E
                        ; ***********************************************************
                        ;
                        ; COMMAND "TRANSMIT X.Y" READS FILE X.Y INTO MEMORY AND
                        ; AFTER THAT TRANSMITS IT TO THE I/O-LINE.
                        ; THE I/O-LINE IS EXPECTED TO BE READY FOR TRANSMISSION.
                        ;
                        ; THIS VERSION IS USED IN MICROMIKKO COMPUTERS.
                        ;     LAST EDITIONS MADE 1982-03-31 BY AR

                        ;TITLE 'TRANSMIT'

                        PAGE 44

                        PAGE
0000 =          BOOT    EQU     0000H   ;SYSTEM REBOOT ADDRESS
0005 =          BDOS    EQU     0005H   ;BDOS ENTRY POINT
005C =          FCBL    EQU     005CH   ;FILE CONTROL BLOCK ADDRESS
0100 =          TPA     EQU     0100H   ;BEGINNING OF TPA
007C =          FCBCR   EQU     FCBL+32 ;CURRENT RECORD NUMBER

                ;       BDOS FUNCTION NUMBERS

0003 =          READER  EQU     03      ;READER INPUT
0004 =          PUNCH   EQU     04      ;PUNCH OUTPUT
0007 =          GETIO   EQU     07      ;GET I/O -BYTE
0008 =          SETIO   EQU     08      ;SET I/O -BYTE
000F =          OPENF   EQU     15
0010 =          CLOSEF  EQU     16
0014 =          READF   EQU     20
001A =          SDMAF   EQU     26
0009 =          PRINT   EQU     9

FF10 =          SIOAD   EQU     0FF10H  ;MPSC DEVICE ADDRESSES
FF12 =          SIOAC   EQU     0FF12H
FF30 =          TMTR    EQU     0FF30H  ;TIMER CHANNEL FOR TRANSMIT
F1F2 =          AMASK   EQU     0F1F2H          ;MASK BYTE ADDRESS FOR MAIN LINE
00FF =          MASK    EQU     0FFH            ;MASK BYTE FOR MAIN LINE

                PAGE
0100            ORG     TPA



0100 314002             LXI     SP,STAKKI       ;SET LOCAL STACK POINTER


                ;       FIRST SET THE MPSC FOR 8 DATA BITS, SET THE BAUD RATE
                ;       AND CHECK THE I/O-BYTE DEFINITION

0103 F3                 DI
0104 3A12FF             LDA     SIOAC
0107 3E05               MVI     A,05H           ;CHANGE MULTIPROTOCOL CIRCUIT PARA-
0109 3212FF             STA     SIOAC           ;METERS TO 8 BITS, EVEN PARITY
010C 3EE1               MVI     A,0E1H
010E 3212FF             STA     SIOAC

0111 2130FF             LXI     H,TMTR          ;SET THE BAUD RATE FOR TRANSMIT
0114 360A               MVI     M,0AH           ;SET FIRST THE LSB OF THE CODE
0116 3600               MVI     M,00H           ;AND THEN THE MSB (000AH=9600 BAUD)

0118 FB                 EI

0119 0E07               MVI     C,GETIO         ;GET I/O-BYTE
011B CD0500             CALL    BDOS
011E 321002             STA     IOBYTE          ;SAVE

0121 E6CF               ANI     11001111B       ;SET PUNCH
0123 5F                 MOV     E,A             ;DEVICE TO MAIN LINE
0124 0E08               MVI     C,SETIO         ;SET I/O -BYTE
0126 CD0500             CALL    BDOS


0129 115C00             LXI     D,FCBL
012C 0E0F               MVI     C,OPENF
012E CD0500             CALL    BDOS            ;OPEN FILE
0131 3C                 INR     A
0132 CAE101             JZ      ERROR1          ;JUMP IF FILE NOT FOUND

0135 AF                 XRA     A
0136 327C00             STA     FCBCR           ;CURRENT RECORD = 0

0139 AF                 XRA     A
013A 321E02             STA     PARBYT          ;RESET PARITY BYTE
013D 214E02             LXI     H,BUFFER
0140 222202             SHLD    DMAAD           ;SET DMA ADDRESS INITIAL VALUE
0143 210000             LXI     H,0
```

```
0145 222002             SHLD    RECCNT          ;RESET RECORD COUNTER

0148 2A2202     LOOP:   LHLD    DMAAD
014C EB                 XCHG
014D 0E14               MVI     C,SDMAF
014F CD0500     --      CALL    BDOS            ;SET DMA ADDRESS

0152 115C00             LXI     D,FCBL
0155 0E14               MVI     C,READF
0157 CD0500             CALL    BDOS            ;READ NEXT RECORD
015A B7                 ORA     A
015B C27201             JNZ     FILEND          ;JUMP IF END OF FILE

015E 2A2002             LHLD    RECCNT
0161 23                 INX     H               ;INCREMENT RECORD COUNTER
0162 222002             SHLD    RECCNT
0165 2A2202             LHLD    DMAAD
0168 118000             LXI     D,80H
016B 19                 DAD     D               ;INCREMENT DMA ADDRESS
016C 222202             SHLD    DMAAD
016F C34901             JMP     LOOP            ;READ NEXT RECORD

0172 115C00     FILEND: LXI     D,FCBL
0175 0E10               MVI     C,CLOSEF
0177 CD0500             CALL    BDOS            ;CLOSE FILE

017A 2A2002             LHLD    RECCNT          ;FILE EMPTY ?????
017D 7C                 MOV     A,H
017E B5                 ORA     L
017F CA8401             JZ      LOOPU           ;JUMP IF FILE EMPTY

0182 7D         FILEXS: MOV     A,L             ;TRANSMIT RECORD COUNT TO THE LINE
0183 CD0902             CALL    TRACHA
0186 CD1402             CALL    PARITY
0189 7C                 MOV     A,H
018A CD0902             CALL    TRACHA
018D CD1402             CALL    PARITY

0190 21AE02             LXI     H,BUFFER        ;TRANSMIT START ADDRESS
0193 AF                 XRA     A
0194 322402             STA     RECLEN          ;RESET RECORD LENGTH COUNTER
0197 EB                 XCHG



0198 EB         LOOPU:  XCHG
0199 7E         LOOP1:  MOV     A,M             ;READ CHARACTER FROM MEMORY
019A CD0902             CALL    TRACHA          ;TRANSMIT IT
019D CD1402             CALL    PARITY          ;COUNT PARITY SUM OVER CHARACTERS
01A0 23                 INX     H
01A1 3A2402             LDA     RECLEN
01A4 3C                 INR     A
01A5 322402             STA     RECLEN
01A8 FE80               CPI     128             ;WHOLE RECORD TRANSMITTED ???
01AA C29901             JNZ     LOOP1           ;JUMP IF NOT

01AD EB                 XCHG
01AE 2A2002             LHLD    RECCNT
01B1 2B                 DCX     H               ;DECREMENT RECORD COUNTER
01B2 222002             SHLD    RECCNT
01B5 AF                 XRA     A
01B6 322402             STA     RECLEN          ;RESET RECORD LENGTH COUNTER
01B9 7D                 MOV     A,L
01BA B4                 ORA     H
01BB C29801             JNZ     LOOPU           ;JUMP IF RECORDS LEFT

01BE 3A1F02             LDA     PARBYT
01C1 CD0902             CALL    TRACHA          ;TRANSMIT PARITY BYTE TO THE LINE

01C4 3A1002             LDA     IOBYTE          ;GET THE OLD IOBYTE DEFINITION
01C7 5F                 MOV     E,A
01C8 0E08               MVI     C,SETIO         ;SET I/O -BYTE
01CA CD0500             CALL    BDOS

01CD 3A1F02             LDA     OMASK           ;SET OLD MASK FOR MAIN LINE
01D0 32F2F1             STA     AMASK

01D3 C30000             JMP     BOOT            ;END OF PROGRAM

01D6 11E701     LOPPU:  LXI     D,MESSU1        ;WRITE ERROR MESSAGES
01D9 0E09       LOPPU1: MVI     C,PRINT
01DB CD0500             CALL    BDOS
01DE C30000             JMP     BOOT

01E1 11F601     ERROR1: LXI     D,MESSU2
01E4 C3D901             JMP     LOPPU1

01E7 0D0A       MESSU1: DB      0DH,0AH
01E9 46494C4520         DB      'FILE EMPTY'
```

```
01E3 0D0A              DB      0DH,0AH
01E5 24                DB      '$'

01E6 0D0A      MESSU2: DB      0DH,0AH
01E8 46494C45204E      DB      'FILE NOT FOUND'
01F6 0D0A              DB      0DH,0AH
01F8 24                DB      '$'

01F9 E5        TRACHA: PUSH    H
01FA F5                PUSH    PSW
01FB 5F                MOV     E,A             ;TRANSMIT ONE CHARACTER
01FC 0E04              MVI     C,PUNCH         ;VIA BDOS CALL: PUNCH
01FE CD0500            CALL    BDOS
0201 F1                POP     PSW
0202 E1                POP     H
0203 C9                RET


0204 47        PARITY: MOV     B,A             ;PARITY ADDS THE CHARACTER IN A
0205 3A1F02            LDA     PARBYT          ;  TO THE PARITY IN 'PARBYT'
0208 A8                XRA     B
0209 321F02            STA     PARBYT
020C C9                RET


               ; THE DATA AREA
               ; -------------

020D           IOBYTE: DS      1               ;I/O -BYTE
020E           DMASK:  DS      1               ;MASK
020F           PARBYT: DS      1
0210           RECCNT: DS      2
0212           DMAAD:  DS      2
0214           RECLEN: DS      1
0215                   DS      40
023D           STAKKI: DS      1
023E           BUFFER: DS      1               ;START OF THE TRANSMIT BUFFER


0246                   END
```

TERMINAL FEATURES

Using the Console

MicroMikko has an integrated console VDU. This console emulates Lear Siegler ADM-3A display terminal, which is one of the most frequently used console terminals in microcomputers. This enables application programs made for ADM-3A to run in MicroMikko without change.

The integrated console consists of a CRT display unit and a keyboard, which are connected directly to the conrol unit of MicroMikko. The display includes 24 normal text rows and one system row.

Keyboard and control codes

In addition to the normal aphanumeric key field the keyboard contains a digit key field, a cursor control field and special function key field. Each key will produce a repeat function if held depressed.

All ASCII control codes (00H - 1FH) can be generated from the alphanumeric key field with the CTRL key pressed. In addition there are some keys that generate directly control codes:

TAB	09H
ESC	1BH
RETURN	0DH
LF	0AH

The cursor control keys generate also control codes according to the ADM-3A specification:

1EH	↖
0AH	↓
08H	↑
08H	←
0CH	→

Function keys

CAPS LOCK-key locks all alphabetic keys to uppercase mode. Nonalphabetic keys operate normally. The normal state is restored by pressing the CAPS LOCK-key another time. The initial state is for uppercase mode.

SHIFT+CTRL+f1 generates a RESET-function. The whole system is reinitiated, i.e. the operation is the same as in a power-on situation.

Special function keys f1-f10 produce as a default value sequences

f1	MENU cr
f2	A:
f3	B:
f4	*.
f5	.*
f6	*.*
f7	DIR_
f8	STAT_
f9	TYPE_
f10	SYSTEM cr

Controlling the CRT

The user can transmit characters to the console CRT using the normal CP/M facilities. These messages can be divided into three different categories:

- normal alphanumeric characters
- control characters
- ESC - sequences

Alphanumeric characters are written into the display at the current cursor position and the cursor is moved correspondingly forward. At the end of line the operation depends on the AUTO NL mode.

Control characters and escape sequences:

ASCII control characters (00H - 1FH) that cause operations are listed below. The rest of the control characters are transparent to the console CRT.

BEL (07H) Causes an audible alarm at the keyboard

BS (08H) Back space. Moves the cursor non destructively one character position to left. BS has no further effect once the first position on the line is reached.

LF Line Feed. Moves the cursor one line downward .The display remains stationary until the bottom line is reached. Then LF causes the entire display to scroll one line upward leaving the cursor positioned on the new bottom line. LF does not return the cursor to the first character position of the new line.

VT (0BH) Cursor Up. Moves the cursor upward one line. No action will occur if the cursor is on the top line of the display.

FF (0CH) Forespace. Moves the cursor non-destructively one character position to the right. From the end of line the cursor is moved to the first column of the next lower line. From the end of the bottom line the cursor is moved to the first column of the same line and the entire display is scrolled upward one line.

CR (0DH) Return Cursor. Moves the cursor to the beginning of the present row.

SI (0FH) Lock Keyboard. Locks the keyboard, disabling all keyboard functions, except RESET.

SO (0EH) Unlock Keyboard. Unlocks the keyboard, restoring all keyboard functions.

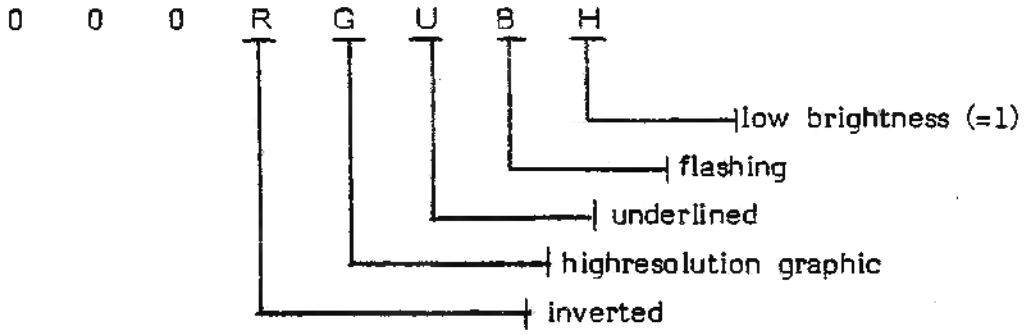
SUB (1AH) Clear Screen. Clear all character position on the display to space. The cursor is moved to HOME position.

ESC (1BH) Starts an escape sequence

RS (1EH) Cursor Home. Moves the cursor to the first column of the top line.

Some controls to the console VDU are transmitted as sequences of characters, the first of which is ESC (1BH). All ESC-sequences used to control the console are listed below.

ESC <00H - 20H> Field attribute definitions. The user can define any combination of flashing, inverted, darkened and underlined fields on the display. Invisible fields can also be defined. Every field attribute definition takes one character position on the display. Field definition continuous to the next field attribute or to the end of display. The second byte of the field definition sequence has the following format in binary representation:



0 0 1 0 0 0 0 0 invisible

For example ESC 11H defines an inverted and darkened field. ESC 00H cancels all field definitions.

ESC ! (1BH 21H) Cursor mode on. Default value. All cursor move commands are available.

ESC " (1BH 22H) Cursor mode off. Cursor control is limited and data is always entered at the bottom line. Cursor move commands Cursor Up and direct cursor positioning are prevented. Cursor Home command moves the cursor to the beginning of the bottom line.

ESC #
(1BH , 23H)

Local Echo mode on
When keys are pressed, characters are both written directly on the display and sent to the CP/M system.

ESC $
(1BH, 24H)

Local Echo mode off.
Default value. Keyed characters are only sent to the CP/M system and they must be echoed back to be written on the display.

ESC = YX
(1BH, 3DH, Y, X)

Direct cursor positioning.
The nect two characters following ESC = sequence represent the absolute line and column (Y and X) coordinates which are used to position the cursor. The line and column nujmbers must be coded to begin at 20H. So the line numbers are 20H - 37H and column numbers 20H - 6FH.

ESC '
(1BH, 27H)

Starts writing on the system row (25th row).
The data between this sequence and a CR is written on the system row starting at the first column of the row. The system roe is automatically cleared, when the writing on it is started. This data can include only normal alphanumeric characters. The cursor is not moved while writing to the system row.

ESC %
(1BH, 25H)

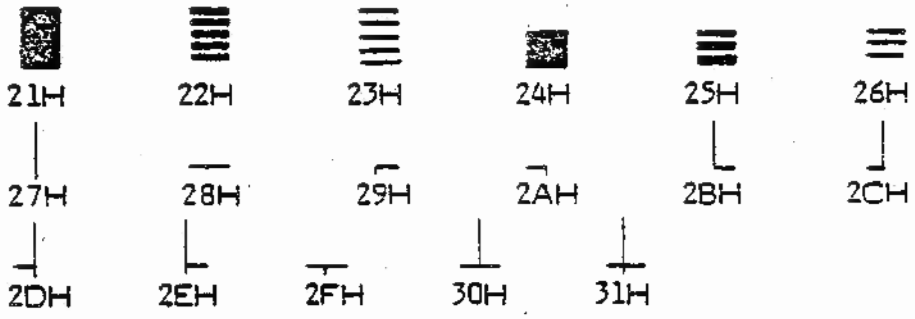
AUTO New Line mode on.
In this mode typing in the last character position will automatically cause the cursor to move to the first position of the next line. Default value is AUTO NL mode on.

ESC &
(1BH, 26H)

AUTO New Line mode off.
In this mode continued typing at the 80th character position changes only the 80th character. Cursor is not moved.

ESC (
(1BH, 28H)

Special character mode on. In this mode ASCII-characters 21H - 31H (!-1) are written as special characters. All other characters are interpreted normally. Special characters and their codes are:



ESC) (1BH, 29H)	Normal character mode on. All characters are displayed normally
ESC * (1B,2A)	Highresolution graphics mode on. The upper part of character generator ROM is selected.

DECIMAL-HEXADECIMAL-ASCII TABLE

Decimal value	Hexadecimal value	ASCII name	Key-board SF/S	Function in MicroMikko
Ø	ØØH	NUL	CTRL+@	
1	Ø1H	SOH	CTRL+A	
2	Ø2H	STX	CTRL+B	
3	Ø3H	ETX	CTRL+C	
4	Ø4H	EOT	CTRL+D	
	Ø5H	ENQ	CTRL+E	
6	Ø6H	ACK	CTRL+F	
7	Ø7H	BEL	CTRL+G	Audible Output
8	Ø8H	BS	←	Backspace, Cursor Left
9	Ø9H	HT	TAB	Tabulate Horizontally
1Ø	ØAH	LF	LF/ ↓	Line Feed/Cursor Down
11	ØBH	VT	↑	Cursor Up
12	ØCH	FF	→	Cursor Right
13	ØDH	CR	RETURN	Carriage Return
14	ØEH	SO	CTRL+N	
15	ØFH	SI	CTRL+O	
	1ØH	DLE	CTRL+P	
17	11H	DC1	CTRL+Q	
18	12H	DC2	CTRL+R	
19	13H	DC3	CTRL+S	
2Ø	14H	DC4	CTRL+T	
21	15H	NAK	CTRL+U	
22	16H	SYN	CTRL+V	
23	17H	ETB	CTRL+W	
24	18H	CAN	CTRL+X	
25	19H	EM	CTRL+Y	
26	1AH	SUB	CTRL+Z	
27	1BH	ESC	ESC	Escape Sequence Start
	1CH	FS	CTRL+Ö	
29	1DH	GS	CTRL+Ä	
3Ø	1EH	RS	↖	Cursor Home
31	1FH	US	CTRL+—	

Decimal	Hex	ASCII	Key	Function
32	20H	SP	(space)	
33	21H	!	!	 Displayed in Special Char.Mode (SCM)
34	22H	"	"	 in SCM
35	23H	#	#	 in SCM
36	24H	$	$	 in SCM
37	25H	%	%	 in SCM
38	26H	&	&	in SCM
39	27H	'	'	 in SCM
40	28H	(	(	in SCM
41	29H	)	)	in SCM
42	2AH	*	*	in SCM
43	2BH	+	+	 in SCM
44	2CH	,	,	in SCM
45	2DH	-	-	 in SCM
46	2EH	.	.	in SCM
47	2FH	/	/	in SCM
48	30H	0	0	in SCM
49	31H	1	1	in SCM
50	32H	2	2	
51	33H	3	3	
52	34H	4	4	
53	35H	5	5	
54	36H	6	6	
55	37H	7	7	
56	38H	8	8	
57	39H	9	9	
58	3AH	:	:	
59	3BH	;	;	
60	3CH	<	<	
61	3DH	=	=	
62	3EH	>	>	
63	3FH	?	?	

Decimal	Hex	ASCII	Key	Function
64	4ØH	@	@	
65	41H	A	A	
66	42H	B	B	
67	43H	C	C	
68	44H	D	D	
69	45H	E	E	
7Ø	46H	F	F	
71	47H	G	G	
72	48H	H	H	
73	49H	I	I	
74	4AH	J	J	
75	4BH	K	K	
76	4CH	L	L	
77	4DH	M	M	
78	4EH	N	N	
79	4FH	O	O	
8Ø	5ØH	P	P	
81	51H	Q	Q	
82	52H	R	R	
83	53H	S	S	
84	54H	T	T	
85	55H	U	U	
86	56H	V	V	
87	57H	W	W	
88	58H	X	X	
89	59H	Y	Y	
9Ø	5AH	Z	Z	
91	5BH	[	Ä	
92	5CH	\	Ö	
93	5DH	]	Å	
94	5EH	↑	^	
95	5FH	←	_	

Decimal	Hex	ASCII	Key	Function
96	60H	`	`	
97	61H	a	a	
98	62H	b	b	
99	63H	c	c	
100	64H	d	d	
101	65H	e	e	
102	66H	f	f	
103	67H	g	g	
104	68H	h	h	
105	69H	i	i	
106	6AH	j	j	
107	6BH	k	k	
108	6CH	l	l	
109	6DH	m	m	
110	6EH	n	n	
111	6FH	o	o	
112	70H	p	p	
113	71H	q	q	
114	72H	r	r	
115	73H	s	s	
116	74H	t	t	
117	75H	u	u	
118	76H	v	v	
119	77H	w	w	
120	78H	x	x	
121	79H	y	y	
122	7AH	z	z	
123	7BH	{	ä	
124	7CH	|	ö	
125	7DH	}	ü	
126	7EH	~	~	␣ in Display
127	7FH	DEL	DEL	■ in Display

4.0 NOTES OF OTHER DOCUMENTATION

1) CP/M Interface Guide says with function 11 status to be 0 or 1. This must be 0 or FF. This is due to an error in orIginal DOS.

2) Introduction to CP/M PIP operation says that PIP is sending an EOF automatically. Not true. Must be commanded like for example OUT:=FILE.ASM, EOF: