Single-chip 8-bit microcontroller family specification

MAB84XX

1 INTRODUCTION

This family data sheet describes the microcontroller core which is common for all members of the MAB84XX family. For complete information of a particular microcontroller, consult both the specific microcontroller data sheet and this family data sheet.

2 FEATURES

- 8-bit CPU
- Up to 8 K bytes ROM
- Up to 128 bytes RAM
- Over 80 instructions, all instructions 1 or 2 cycles
- Up to 20 quasi bi-directional I/O port lines
- 8-bit programmable timer/event counter
- 3 single-level vectored interrupts (external, timer/counter and serial I/O)
- 2 Test inputs: T0 (may also be used as an interrupt) and T1 (may also be used as an input to an 8-bit counter)
- Serial I/O interface
- On-chip oscillator

The family is well supported with:

- Cross assemblers
- In-circuit emulation tools
- Window debugger
- Piggy-back versions for prototyping.

3 GENERAL DESCRIPTION

The MAB84XX single-chip 8-bit microcontroller family is fabricated in NMOS. Members contain on-chip mask programmable program ROM (up to 6 K bytes, except the ROM-less version which can address up to 8 K bytes of external ROM), up to 128 bytes RAM, an interrupt input, a test input (directly testable), a serial I/O and general purpose I/O lines. The instruction set is based on that of the MAB8048. A number of the MAB84XX family members have similar CMOS counterparts in the PCF84CXXX microcontroller family.

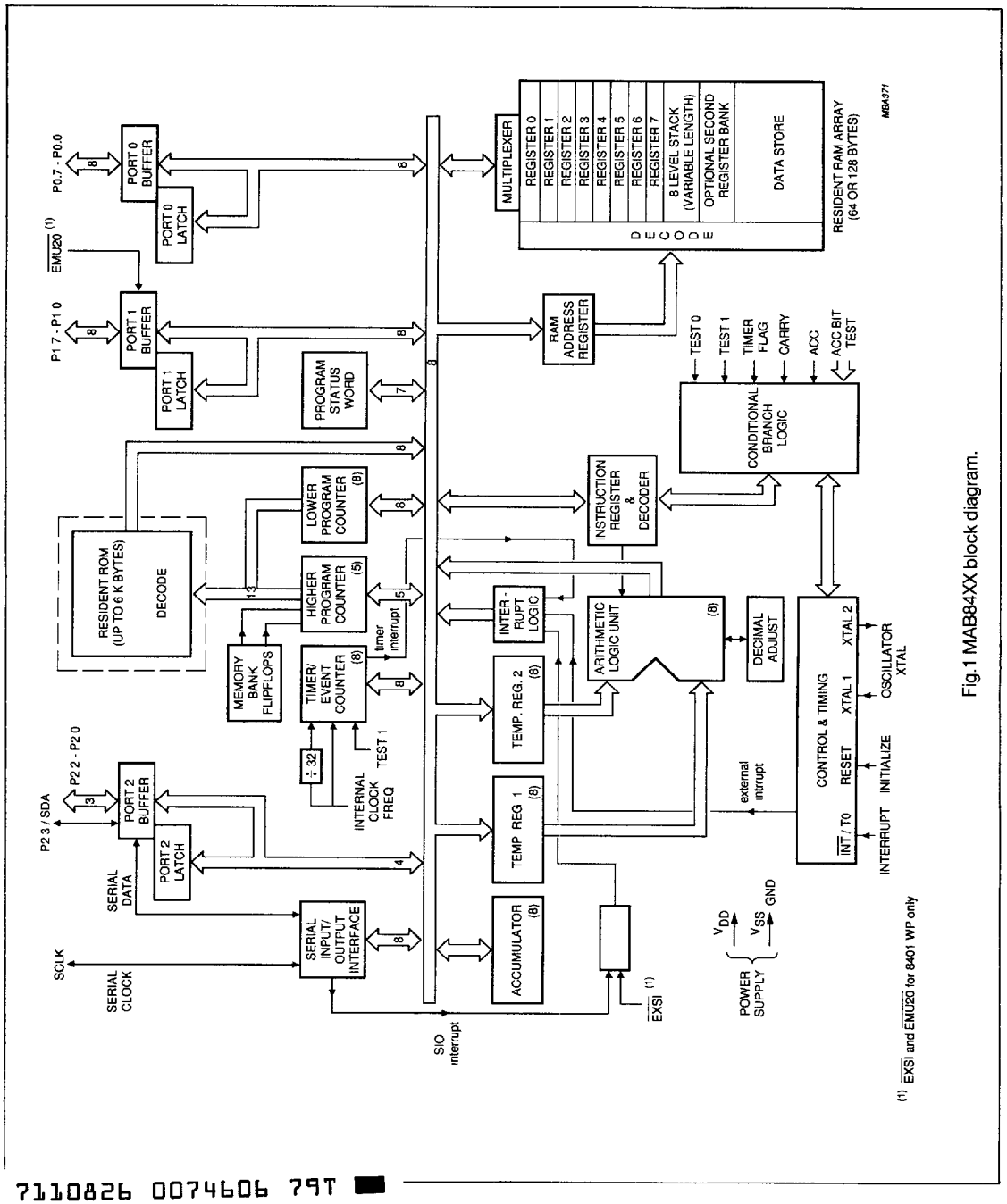


Fig.1 MAB84XX block diagram.

4 CENTRAL PROCESSING UNIT (CPU)

The CPU performs arithmetic, logical and program control functions. The CPU consists of:

- **ALU** (arithmetic and logic unit) which performs the following operations:
  arithmetic operations: ADD, INCREMENT, DECREMENT, ROTATE. The DA A, SWAP A, and XCHD instructions and a half carry flag simplify BCD arithmetic and the handling of nibbles.
  logical operations: OR, AND, EXCLUSIVE-OR, and COMPLEMENT.
- **ACCUMULATOR**; in most operations this is the source or destination register.
- **Program status word**; indicates the status of the microcontroller.
- **Program counter**; supplies a 13-bit address to the program memory.
- **Instruction decoder**; decodes the instructions and supplies control signals to several parts of the microcontroller.
- **Control and timing**; contains the control, oscillator and timing circuitry. A machine cycle consists of 10 states, each state contains 3 XTAL clock periods. Thus, one machine cycle consists of 30 XTAL clock periods. All instructions take 1 or 2 machine cycles.

5 MEMORY AND REGISTERS

5.1 Program memory

Members of the MAB84XX family contain 1 K, 2 K, 4 K, or 6 K bytes of on-chip read-only memory (ROM); there are also ROM-less versions which can address up to 8 K bytes of external ROM. Each location is directly addressable by the program counter. The program memory is mask-programmed at the factory. Figure 2a shows the program memory map.

Four program memory locations are of special importance:

- Location 0: first instruction to be executed after the controller is reset.
- Location 3: first instruction of an external interrupt ($\overline{INT}$/T0) service routine.
- Location 5: first instruction of a serial I/O interrupt service routine.
- Location 7: first instruction of a timer/event counter interrupt service routine.

Program memory is arranged in banks of 2 K bytes. Memory banks are preselected by the SEL MB instructions. Memory bank boundaries can only be crossed by using the unconditional jump (JMP) or subroutine call (CALL) instructions, after the appropriate memory bank has been selected. The program memory is further divided into 'pages', each of 256 bytes. Page boundaries can not be crossed by conditional jumps and indirect jump (JMPP) instructions.

The MOVP A,@A instruction permits efficient table look-up from the current ROM page.

5.2 Data memory

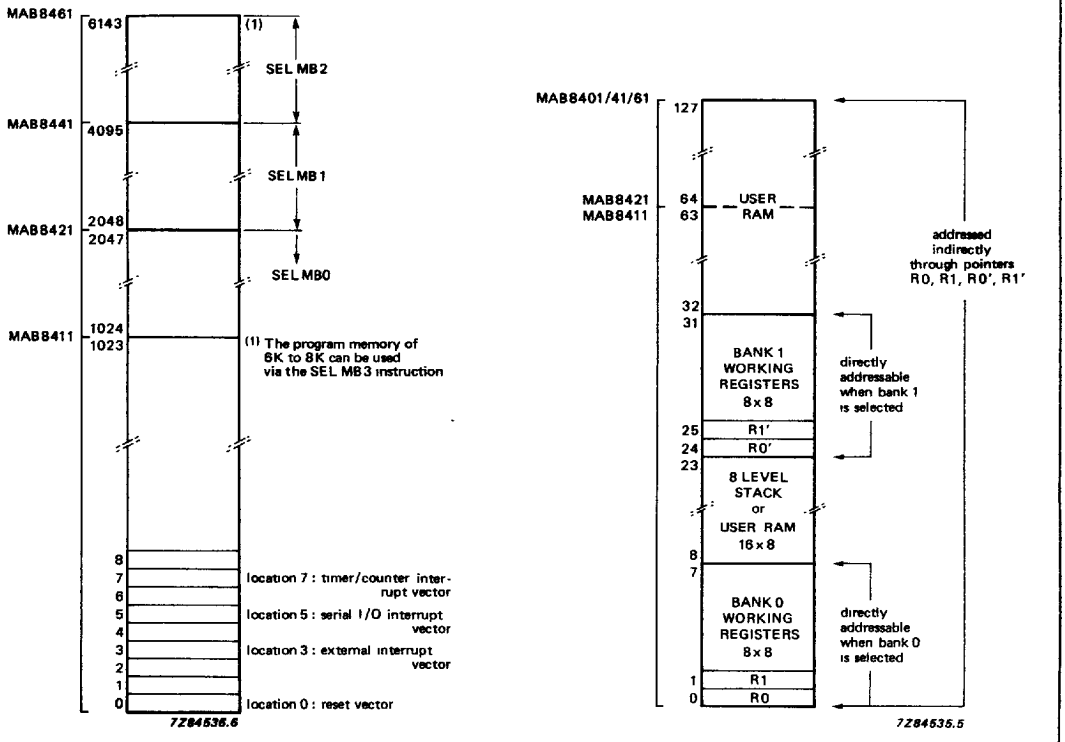
Members of the MAB84XX family contain up to 128 bytes of random access data memory (RAM). All locations are indirectly addressable using RAM pointer registers; up to 16 designated locations are directly addressable (register banks 0 and 1). Another 16 bytes are designated to an 8-level program counter stack addressed by a 3-bit stack pointer. Figure 2b shows the data memory map.

5.2.1 Registers R0 to R7

Registers R0 to R7 are directly addressable by the direct register instructions. Ease of addressing, and a minimum requirement of instruction bytes to manipulate their contents, makes these locations suitable for storing frequently addressed intermediate results.

Executing the SEL RB0 (SELect Register Bank 0) instruction designates R0-R7 to data memory locations 0-7. Executing the SEL RB1 instruction designates R0-R7 to data memory locations 24-31. This second register bank may be used as an extension of the first, or it may be reserved for use during interrupt service routines leaving the first bank available for the main program.

The first 2 locations of each bank contain the RAM pointer registers R0, R1 and R0', R1' which indirectly address all data memory locations. Every data memory location can operate as a loop counter when used with the decrement and test instruction DJNZ @Rr,addr.



(a) Program memory map (b) Data memory map

Fig.2 MAB84XX memory and register map.

5.2.2 Stack

Locations 8 to 23 of the data memory are designated to an 8-level program counter stack (2 locations per level). See Fig.3. A 3-bit stack pointer (SP) points to the next free location on the stack. After a RESET, the SP contents are 000 and the SP points to data memory locations 8 and 9.

During a subroutine CALL or interrupt, the contents of the 13 bit program counter (PC) and bits 4, 6 and 7 of the program status word (PSW) are first transferred to the stack. The SP is then incremented. During a RET or RETR instruction, the 13 bit program counter is restored and the stack pointer is decremented. Only the RETR instruction transfers the saved PSW bits to the PSW.

Nesting of subroutines and interrupt service routines can continue to a depth of 8 levels without overflowing the stack. When an overflow occurs the deepest address stored (locations 8 and 9) will be overwritten and lost since the stack pointer overflows from 111 to 000. The stack pointer also underflows from 000 to 111.

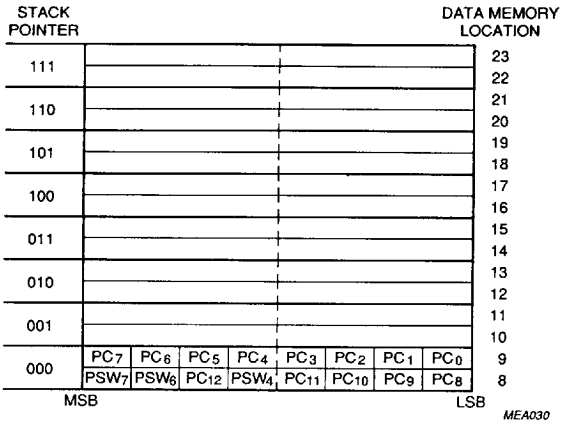


Fig.3 Program counter stack.

5.2.3 User RAM

Data memory locations 32 to 127 are designated as user RAM and are indirectly addressable with the RAM pointers R0, R1 or R0', R1'. Unused register and stack locations are also available as user RAM and are addressed in the same way.

6 PROGRAM COUNTER

The 13-bit program counter can address up to 8 K bytes of ROM (see Fig.4). The least significant 11 bits (PC0 to PC10) are auto-incrementing. The two most significant bits (PC11 and PC12) are loaded with the contents of two internal flip flops (MBFF0 and MBFF1 respectively) when a JMP or CALL instruction is executed. The contents of these two internal flip-flops are altered using the SEL MB instructions. During an interrupt service routine PC11 and PC12 are forced to logic 0.

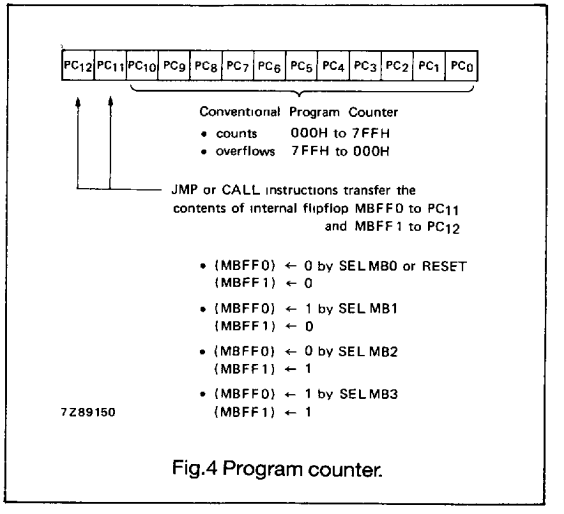


Fig.4 Program counter.

7 PROGRAM STATUS WORD

The program status word (PSW) is an 8-bit CPU register which stores information about the current status of the microcontroller (see Fig.5). All bits can be read using the MOV A,PSW instruction. Only the PS bit can be written using the MOV PSW,A instruction. Table 1 shows the function of the PSW bits and when they are affected.

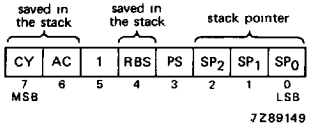


Fig.5 Program status word.

Table 1 Program Status Word.

BIT	NAME	FUNCTION	AFFECTED BY
7	CY	Carry, signals accumulator overflow	ADD, ADDC, DA, RLC, RRC, CLR C and CPL C instructions
6	AC	Auxiliary Carry, half carry	ADD and ADDC instructions
5	-	Not used, always 1 when read	
4	RBS	Register Bank Select 0: select register bank 0 1: select register bank 1	SEL RB instructions
3	PS	Timer Prescaler Select 0: prescaler selected (÷32) 1: prescaler not selected (÷1)	MOV PSW,A instruction
2 1 0	SP2 SP1 SP0	Stack Pointer bits 2-0	CALL, RET, RETR instructions and interrupts

8 CONDITIONAL JUMP LOGIC

The conditional jump logic enables several conditions to be tested by the user's program. Table 2 lists the conditional jump instructions which may be used to change the program sequence.

The DJNZ instruction decrements a designated register or data memory location and jumps if the contents are not zero. This instruction is useful for looping control.

Table 2 Conditional jumps.

TEST	JUMP CONDITION	JUMP INSTRUCTION
accumulator	all bits zero any bit non-zero	JZ JNZ
accumulator bit test	1	JB0 to JB7
carry flag	1 0	JC JNC
timer overflow flag	1 0	JTF JNTF
test input T0	1 0	JT0 JNT0
test input T1	1 0	JT1 JNT1
register	non-zero	DJNZ Rr
data memory location	non-zero	DJNZ @Rr

9 INTERRUPT LOGIC

The MAB84XX family handles external, serial I/O, and timer/event counter interrupts. The interrupt mechanism is single level; an executing interrupt service routine can not be interrupted by other interrupts. If several interrupt requests are detected simultaneously, they are serviced in order of priority (see Table 3). An interrupt request will only be serviced if the interrupt is enabled; each interrupt type has individual enable and disable instructions (see Table 3).

Before an interrupt is serviced, execution of the current instruction is first completed. The contents of the program counter, and bits 4, 6 and 7 of the program status word are then saved on the program counter stack, and a CALL to the corresponding interrupt vector is executed (see Table 3). The maximum interrupt latency time is 4 machine cycles. The interrupt service routine must be terminated by the RETR (return and restore) instruction. At least one instruction in the main program will then be executed before another interrupt service routine is serviced.

Table 3 Interrupts.

INTERRUPT			INSTRUCTION	
PRIORITY	**TYPE**	**VECTOR LOCATION**	**ENABLE**	**DISABLE**
1 (highest)	external	003 (ROM)	EN I	DIS I
2	serial I/O	005 (ROM)	EN SI	DIS SI
3 (lowest)	timer/event counter	007 (ROM)	EN TCNTI	DIS TCNTI

Notes on interrupt service routines

- While an interrupt service routine is in progress, the two most significant bits of the program counter are frozen at zero.
  Therefore: interrupt service routines and any subroutines called within interrupt service routines must reside (entirely) in memory bank 0.
  Within an interrupt service routine and within any subroutines called within an interrupt service routine, other memory banks cannot be selected and SEL MB instructions must not be used.
- Subroutines and nested subroutines called within an interrupt service routine must all end with RET. RETR would clear the interrupt in progress flag and further pending interrupts would then interfere with the interrupt service routine in progress.

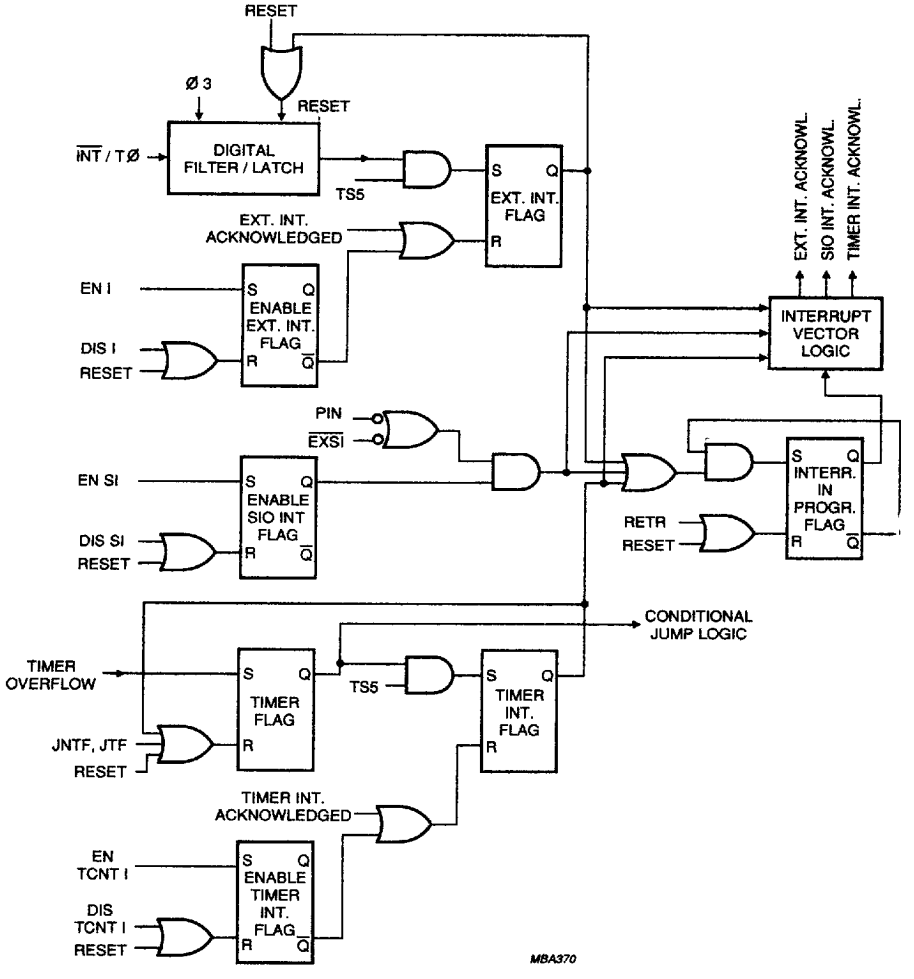


Fig.6 Interrupt logic.

Notes:

1. When the Interrupt In Progress (IIP) flag is set, other interrupts are latched but ignored until the RETR instruction is executed.
2. When the timer/counter interrupt flag (TF) has been enabled, the JTF and JNTF instructions will always find the flag at logic 0; this is because the timer flag is set at time slot 3 and reset at time slot 5 of the same cycle.

9.1 External interrupt

A HIGH-to-LOW transition on the $\overline{INT}$/T0 pin generates an external interrupt request which is always latched in a digital filter/latch (see Fig.6). To ensure that the transition is latched, the LOW state must exceed 7 XTAL clock periods after a HIGH state of more than 4 XTAL clock periods. When the External Interrupt Flag (EIF) is set, it resets the digital filter/latch (see Fig.6). If the external interrupt is enabled and no interrupt service routine is in progress, the external interrupt will be serviced. Simultaneously, the hardware removes the latched interrupt request.

During the forced CALL to the external interrupt vector location, EIF is reset and the digital filter/latch is re-enabled; the Interrupt In Progress (IIP) flag bit is also set so that other interrupts are latched but ignored until the RETR instruction is executed. After the RETR instruction has been executed, latched interrupts will be serviced in order of priority. Execution of a DIS I (disable external interrupt) instruction cancels a pending external interrupt request which has been latched in the EIF. An interrupt request which is latched in the digital filter/latch after the external interrupt has been disabled can't be cancelled.

9.2 Serial I/O interrupt

The serial I/O interrupt may be requested by the serial I/O interface. An interrupt request from the SIO interface will force the PIN interrupt flag to logic 0. If the serial I/O interrupt is enabled and no interrupt service routine is in progress, the serial I/O interrupt will be serviced. The interrupt flag is NOT automatically reset to the inactive state (by hardware) when a serial I/O interrupt is serviced; this must be done by user software.

9.3 Timer/event counter interrupt

A timer/event counter overflow generates a timer/event counter interrupt request. This interrupt request is always latched. If the timer/event counter interrupt has been enabled and if no interrupt service routine is in progress, the timer/event counter interrupt will be serviced. Simultaneously, the hardware removes the latched interrupt request. Execution of a DIS TCNTI (disable timer/event counter interrupt) instruction cancels a pending timer/event counter interrupt request which has been latched in the Timer Interrupt Flag.

9.4 Interrupt operation examples

Figures 7 to 10 show examples of how the interrupt mechanism operates.

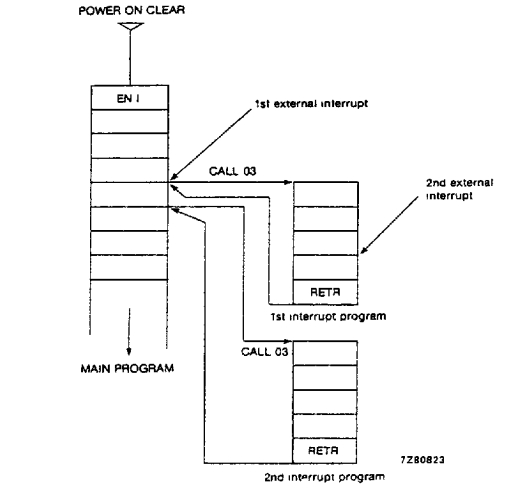


Fig.7 An external interrupt which occurs while another interrupt service routine in being executed remains latched until the executing routine is complete.

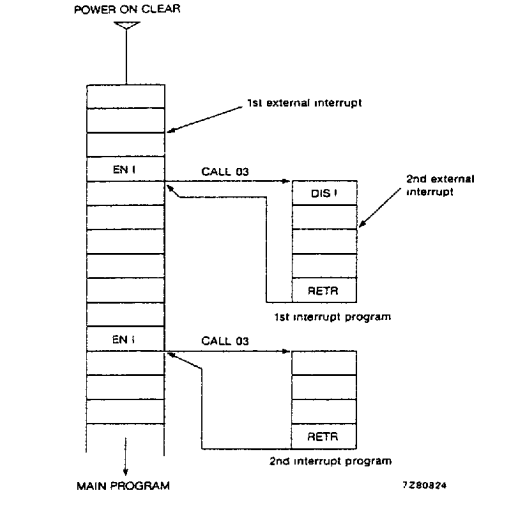


Fig.8 External interrupts are postponed until enabled.

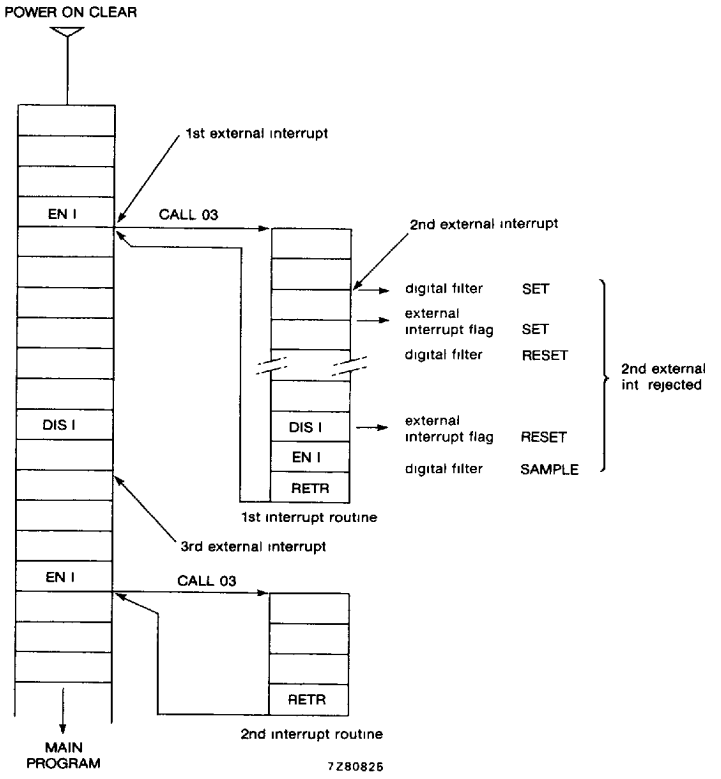


Fig.9 A previous external interrupt is cleared.

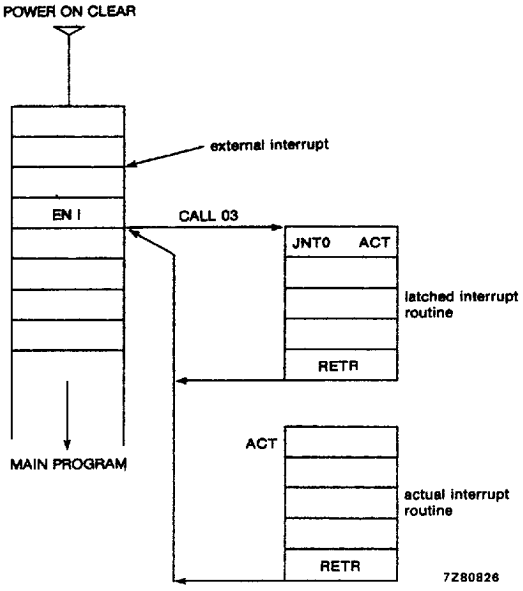


Fig.10 Detection of previous and actual external interrupt.

10 TIMER/EVENT COUNTER

The Timer/event counter (see Fig.11) can operate either as a timer or as an event counter.

- **Timer mode**; depending on the state of the PS-bit in the program status word, the internal 8-bit up-counter is incremented every machine cycle (PS = 1) or every 32 machine cycles (PS = 0).
- **Counter mode**; the internal 8-bit up-counter is incremented on every LOW-to-HIGH transition on pin T1. The HIGH state of T1 must exceed 4 XTAL clock periods after a LOW state of more than 4 XTAL clock periods. The maximum count rate is one increment per machine cycle.

When the 8-bit up-counter overflows, the timer overflow flag is set. If the timer/event counter interrupt is enabled, the timer interrupt flag (see Fig.6) is also set and the timer overflow flag is reset again in the same machine cycle. Table 4 details the instructions that control the timer/event counter. Testing the timer overflow flag, using the timer JTF (jump if timer flag = 1) or JNTF instructions, also clears the timer overflow flag. Starting the timer, using the STRT T instruction, also clears the pre-scaler. Reading the 8-bit up-counter does not disturb the counting process.

Table 4 Timer/event counter control.

FUNCTION	TIMER MODE	COUNTER MODE
clear	MOV T,A (A)=0 or RESET	MOV T,A (A)=0 or RESET
preset	MOV T,A	MOV T,A
start	STRT T	STRT CNT
stop	STOP TCNT or RESET	STOP TCNT or RESET
test	JTF/JNTF	JTF/JNTF
read	MOV A,T	MOV A,T

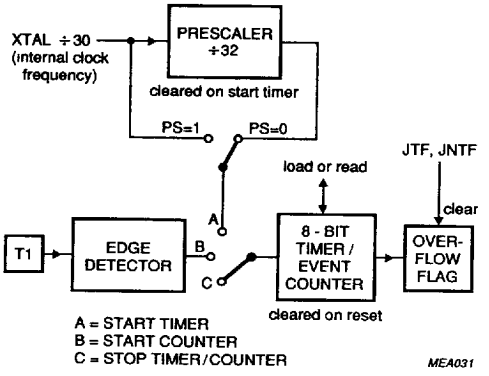


Fig.11 Timer/event counter.

11 I/O

Members of the MAB84XX family have up to 23 I/O lines arranged as follows:

- two 8-bit parallel ports
- one 4-bit parallel port
- a serial I/O which consists of the following two lines:
  - P2.3/SDA; data line shared with port line P2.3
  - SCLK, dedicated clock line
- $\overline{INT}$/T0, an external interrupt/ test input
- T1, a test input which may also be used as an input to the counter/timer

11.1 Parallel I/O ports

Up to three parallel ports are controlled by dedicated port instructions.

These ports can have up to 20 I/O port lines arranged as:

- Port 0: 8-bit parallel port (P0.0 to P0.7)
- Port 1: 8-bit parallel port (P1.0 to P1.7)
- Port 2: 4-bit parallel port (P2.0 to P2.3).

Output data written to a port is latched and remains unchanged until rewritten. Input data is not latched and must therefore remain present until read by an input instruction. Fig.12 shows the timing diagram for all ports using IN, OUTL, ANL and ORL instructions. For the OUTL instruction, data changes on time slot 7 of cycle 1. For the ANL and ORL instructions, port data changes on time slot 7 of cycle 2.

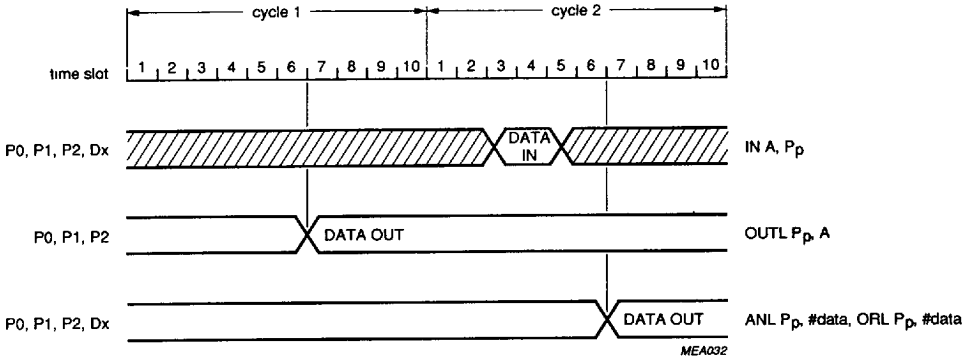


Fig.12 Port timing.

Input port lines are fully TTL compatible; output port lines can drive one TTL load. The high drive capability of Port 1 was intended to enable the 8 Port 1 output lines to drive 8 LEDs.

Three I/O mask options make it possible for parallel I/O port lines to be individually configured as follows:

- **Option 1** STANDARD I/O; quasi-bidirectional I/O (see Fig.13) consisting of a push-pull output with a weak pull-up transistor (TR3). If the output latch contains a logic 1, then the port line is pulled up to V_{CC} via TR3. TR3 provides sufficient current for a HIGH level to one TTL input, yet can be pulled LOW by an external TTL or CMOS device, thus allowing the same port line to be used for both input and output. When used as an input, the port latch must be in a logic one state. When a logic 0 is written to the port latch, a low impedance transistor (TR1) in turned on to provide TTL current sinking capability. To provide a fast logic 0 to logic 1 transition, TR2 is active during 1 XTAL clock cycle when a logic 1 is written to the port latch.
- **Option 2** OPEN DRAIN I/O WITH PULL-UP; open drain output with pull-up transistor, TR3 (see Fig.14).
- **Option 3** OPEN DRAIN I/O WITHOUT PULL-UP; open drain output without pull-up transistor (see Fig.15). Application as an output requires the connection of an external pull-up resistor.

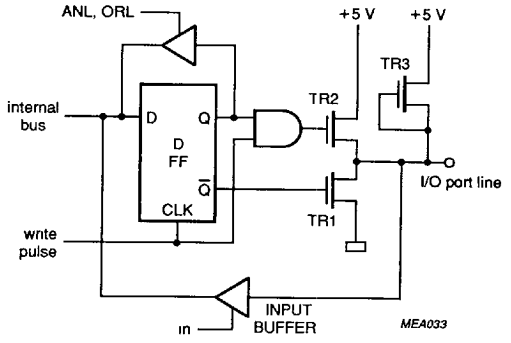


Fig.13 Standard push-pull I/O with pull-up transistor (option 1).

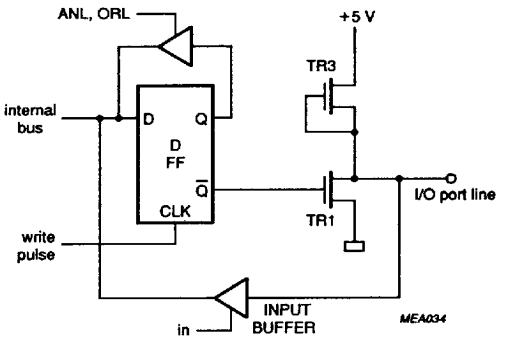


Fig.14 Open drain I/O with pull-up transistor (option 2).

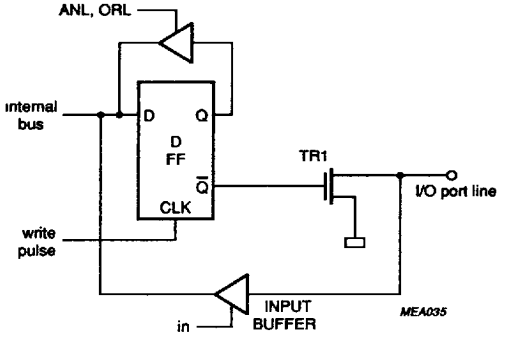


Fig.15 Open drain I/O without pull-up transistor (option 3).

11.2 Test inputs T1 and $\overline{INT}$/T0

The T1 input line can be used as:
- a test input for jump instructions JT1 and JNT1
- an input for zero voltage cross-over detection
- an external input to the event counter.

A pull-up resistor can be provided as a ROM mask option; this is useful when the input is from a switch or a standard TTL output.

When used as a test input, the JT1 and JNT1 instructions test the T1 pin for logic 1 and logic 0 respectively.

The T1 input has a self biasing circuit which can detect when an AC signal crosses zero (see Fig.16).

A LOW-to-HIGH transition on the T1 input increments the timer/event counter when the timer/event counter is in the counter mode.

When used in conjunction with the timer/event counter interrupt, zero cross-over detection is useful in thyristor control of power equipment.

The $\overline{INT}$/T0 input line can be used as:
- a test input for jump instructions JT0 and JNT0
- an external interrupt input.

11.3 Serial I/O

The MAB84XX on-chip serial I/O (SIO) hardware enables MAB84XX family members to be interconnected via the two-line serial I^2C bus. The SIO allows two or more devices to communicate without interrupting other devices on the bus. This is achieved by allocating a specific 7-bit address to each device on the bus. Each device only reacts to messages prefixed with the specific address of the device or the general call address. Address recognition is handled by the SIO hardware, and the microcontroller is interrupted only when a valid address has been detected. The SIO hardware thus saves considerable CPU resources and memory requirements (compared to a software serial interface).

When address recognition is not required (e.g. a configuration with only two microcontrollers connected to the serial bus), direct data transfer without addressing can be performed. In multi-master systems, an automatic arbitration procedure stops two or more devices from transmitting simultaneously.

For more information on the SIO, see the chapter 'SERIAL I/O'.

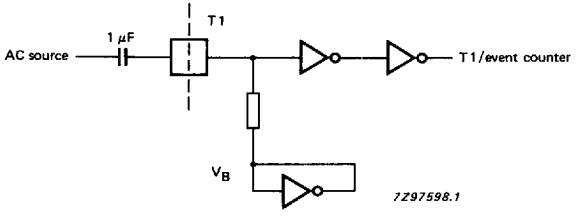


Fig.16 Self biasing circuit.

12 OSCILLATOR

The oscillator circuit must be completed by connecting one of the following timing elements between the XTAL1 and XTAL2 pins:

- a quartz crystal
- a ceramic resonator
- an inductor.

Two external load capacitors are also required.

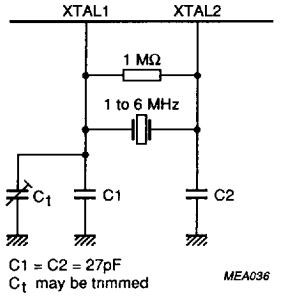
If a quartz crystal is used as the timing element, the crystal must operate in the fundamental mode of vibration. The clock may also be supplied by an external source. In this case the rise and fall times of the external clock (t_r, t_f) must be less that 10 ns. A pull-up resistor is required if the clock source is a TTL device. Table 5 gives the recommended L and C values for various oscillator frequencies when the timing element is an inductor.

Table 5 LC oscillator timing.

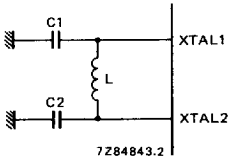
FREQUENCY (MHz)	C1 = C2 (pF)	L (μH)
3.0	33	100
4.0	33	56
4.4	33	47
5.0	33	33
6.0	33	22

Oscillator start-up time depends on the external timing element. The start-up time of a quartz crystal is several milliseconds. The start-up time of a ceramic resonator is tenths of a millisecond.

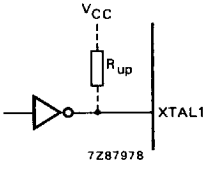
The XTAL clock may also be supplied by an external clock signal at pin XTAL1. In this case, XTAL2 is not connected.



(a) Crystal or ceramic resonator



(b) Inductor



(c) External

Fig.17 Oscillator circuit with various timing elements.

13 RESET

A reset signal initializes the microcontroller to a defined state.

13.1 Reset sequence

To ensure a correct reset, the reset signal at the RESET pin must be HIGH for at least 2 machine cycles after the power supply and clock have stabilized. The HIGH level at the RESET pin:

- sets the program counter to zero
- selects memory bank 0 and register bank 0
- sets the stack pointer to zero (000), pointing to RAM addresses 8 and 9
- resets interrupt flags (external, timer/event counter and serial I/O) to the inactive state
- disables interrupts (external, timer/event counter and serial I/O)
- stops the timer/event counter, then sets it to zero
- sets the timer prescaler to divide by 32
- resets the timer overflow flag
- sets all ports latches to logic one (input mode), except P2.3/SDA which is high impedance
- sets the serial I/O to the slave receiver mode and disables the serial I/O.

A reset does not affect the data memory contents.

The external power-on-reset circuit should consist of a 1 μF capacitor connected between V_{CC} and the RESET pin (see Fig.18); a diode may also be connected between the RESET pin and ground to ensure a correct reset if the supply voltage falls momentarily. Alternatively the device can be reset by an external TTL compatible signal (see Fig.19). Figure 20 shows typical input characteristics.

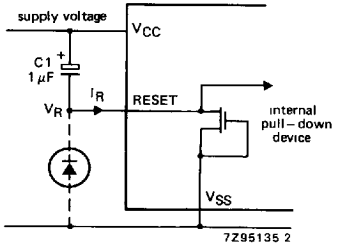


Fig.18 Power-on-reset.

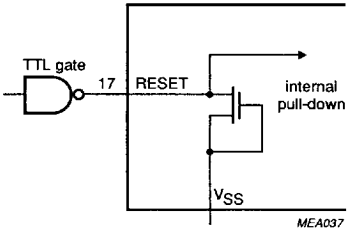


Fig.19 External reset.

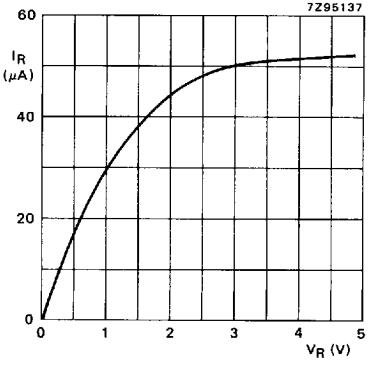


Fig.20 Typical input characteristics.

14 INSTRUCTION SET

The MAB84XX family instruction set consists of over 80 one and two byte instructions. Program code efficiency is high because all RAM locations and all ROM locations on a 256 byte page require only a single byte address.

Figure 21 shows the instruction map and Table 6 describes the symbols that are used.

Table 6 Symbols and definitions.

SYMBOL	DESCRIPTION
A	accumulator
addr	program memory address
Bb	bit designation (b = 0...7)
C	carry bit (bit CY)
CNT	event counter
data	8-bit immediate data
I	interrupt
MBn	memory bank (n = 0...3)
MBFFn	memory bank flip-flop (n = 0, 1)
P	mnemonic for 'in-page' operation
PC	program counter
Pp	port designation (p = 0, 1, 2)
PSW	program status word
RB	register bank
RBS	register bank select flag
@Rr	8-bit address register (r = 0, 1)
Rr	8-bit register (r = 0...7)
Sn	serial I/O register (n = 0, 1, 2)
SP	stack pointer
T	timer
TCNT	timer/event counter
TF	timer flag
T0, T1	test 0 and test 1 inputs
#	immediate data prefix
@	indirect address prefix
(X)	contents of X
((X))	contents of location addressed by X
←	is replaced by
↔	is exchanged with

first hexadecimal character of opcode ↓ | second hexadecimal character of opcode →

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP			ADD A #data	JMP page 0	EN I	JNTF addr	DEC A		IN A,Pp 0 1 2			MOV A Sn 0 1			
1	INC @ Rr 0 1		JB0 addr	ADDC A,#data	CALL page 0	DIS I	JTF addr	INC A		INC Rr 0 1 2 3 4 5 6 7						
2	XCH A, @Rr 0 1			MOV A, #data	JMP page 1	EN TCNTI	JNT0 addr	CLR A		XCH A,Rr 0 1 2 3 4 5 6 7						
3	XCHD A, @Rr 0 1		JB1 addr		CALL page 1	DIS TCNTi	JT0 addr	CPL A		OUTL Pp,A 0 1 2			MOV Sn,A 0 1 2			
4	ORL A, @Rr 0 1		MOV A, T	ORL A, #data	JMP page 2	STRT CNT	JNT1 addr	SWAP A		ORL A,Rr 0 1 2 3 4 5 6 7						
5	ANL A @Rr 0 1		JB2 addr	ANL A, #data	CALL page 2	STRT T	JT1 addr	DA A		ANL A,Rr 0 1 2 3 4 5 6 7						
6	ADD A @Rr 0 1		MOV T, A		JMP page 3	STOP TCNT		RRC A		ADD A,Rr 0 1 2 3 4 5 6 7						
7	ADDC A, @Rr 0 1		JB3 addr		CALL page 3			RR A		ADDC A,Rr 0 1 2 3 4 5 6 7						
8				RET	JMP page 4	EN SI				ORL Pp,#data 0 1 2						
9	OUTL PO,A		JB4 addr	RETR	CALL page 4	DIS SI	JNZ addr	CLR C		ANL Pp,#data 0 1 2			MOV Sn,#data 0 1 2			
A	MOV @ Rr,A 0 1			MOVP A,@A	JMP page 5	SEL MB2		CPL C		MOV Rr,A 0 1 2 3 4 5 6 7						
B	MOV @Rr, #data 0 1		JB5 addr	JMPP @A	CALL page 5	SEL MB3				MOV Rr,#data 0 1 2 3 4 5 6 7						
C	DEC @Rr 0 1				JMP page 6	SEL RB0	JZ addr	MOV A,PSW		DEC Rr 0 1 2 3 4 5 6 7						
D	XRL A, @Rr 0 1		JB6 addr	XRL A,#data	CALL page 6	SEL RB1		MOV PSW,A		XRL A,Rr 0 1 2 3 4 5 6 7						
E	DJNZ @ Rr,addr 0 1				JMP page 7	SEL MB0	JNC addr	RL A		DJNZ Rr,addr 0 1 2 3 4 5 6 7						
F	MOV A, @Rr 0 1		JB7 addr		CALL page 7	SEL MB1	JC addr	RLC A		MOV A,Rr 0 1 2 3 4 5 6 7						

MBA369

Fig.21 Instruction map.