8 Bit Development Microcomputer

FEATURES

- *PIC1670 microcomputer with ROM removed*
- Useful for engineering prototyping of PIC applications
- PIC ROM address & data lines brought out to pins
- HALT pin for single stepping or stopping program execution
- User programmable via external memory
- 64 8-bit RAM registers
- Arithmetic Logic Unit
- User defined TTL-compatible Input and Output lines
- Real Time Clock/Counter
- Self-contained oscillator
- Access to RAM registers inherent in instruction
- Power supply operating range (4.5V to 5.5V)

DESCRIPTION

The PIC1665 development microcomputer is an MOS/LSI device containing RAM, I/O, and a central processing unit on a single chip.

The PIC1665 MOS/LSI device is functionally identical to the PIC1670 microcomputer except that the ROM is removed and the ROM address and data lines are brought out, requiring a 64-pin package. The addition of a HALT pin gives the user the ability to stop as well as single-step the chip.

The external ROM can contain a customer-defined program using the PIC's powerful instruction set to specify the overall functional characteristics of the device. The 8-bit input/ouput registers provide latched lines for interfacing to a limitless variety of applications.

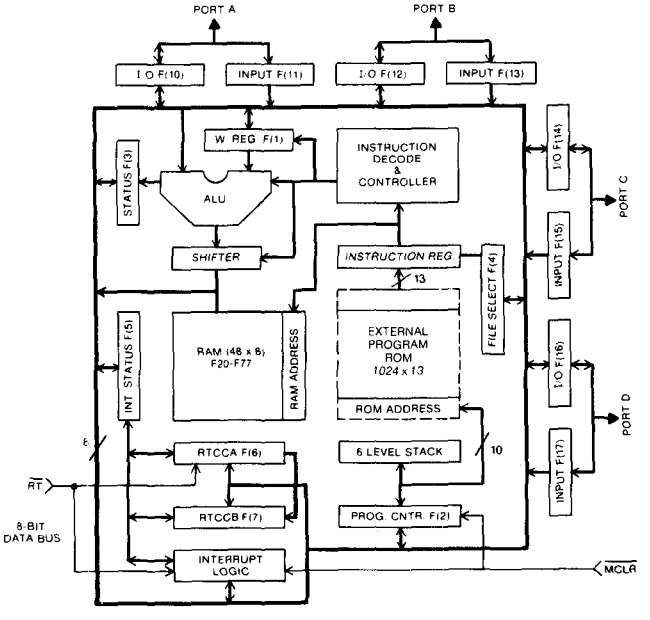
The 13-bit instruction word format provides a powerful yet easy to use instruction repertoire emphasizing single bit manipulation as well as logical and arithmetic operations using bytes.

The PIC1665 is fabricated with N-Channel Si-gate technology resulting in a high performance product with proven reliability and production history. Only a single power supply is required for operation, and an on-chip oscillator provides the operating clock with an external crystal, ceramic resonator or LC network to establish the frequency. Inputs and outputs are TTL-compatible.

Extensive hardware and software support is available to aid the user in developing this application program and to verify performance before committing to mask tooling. Application notes and sample programs are used to develop programs which can then be assembled into machine language using PICAL, eliminating the burden of coding with ones and zeros. PICAL is available in a Fortran IV version that can be run on many popular computer systems. Once the application program is developed several options are available to insure proper performance. The PFD1020 Field Demo System is available containing a PIC1665 with sockets for erasable PROMs. Finally, the PICES II (PIC In-Circuit Emulation System) provides the user with stand-alone emulation and debugging operation or operation as a peripheral to a larger computer system. Easy program debugging and changing is facilitated because the user's program is stored in RAM.

MICROCOMPUTER

PIC1665 BLOCK DIAGRAM



ARCHITECTURAL DESCRIPTION

The firmware architecture of the PIC1665 microcomputer is based on a register file concept with simple yet powerful instruction commands designed to optimize the code for bit, byte, and register transfer operations. The instruction set also supports computing functions as well as these control and interface functions.

Internally, the functional blocks of the PIC1665 are connected by an 8-bit bidirectional bus: the 64 8-bit registers of which the first 16 are special purpose, an Arithmetic Logic Unit, and a user defined external PROM composed of 1024 x 13 bit words. The register file is divided into two functional groups: operational registers and general purpose registers. The first sixteen are the operational registers and they include the Real Time Clock/Counters A and B, four I/O registers, two Status registers, a Program Counter and a File Select Register. The general purpose registers are used for data and control information under command of the instructions.

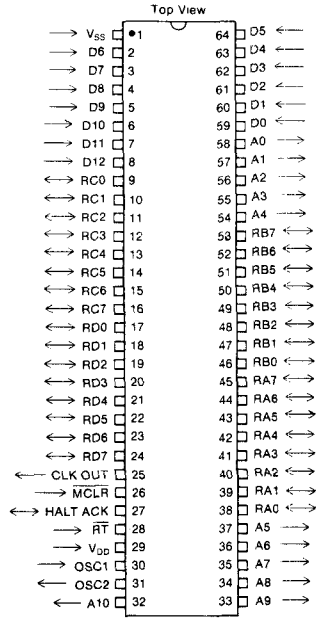
The Arithmetic Logic Unit contains one temporary working register (W Register), an adder, and hardware for decimal adjust. Manipulation between data in the working register and any other register can be performed.

The external PROM contains the user defined application program and is supported by an instruction decoder and instruction register. Sequencing of microinstructions is controlled via the Program Counter (PC) which automatically increments to execute in-line programs. The Program Counter is modified by bit test, jump, call or branch instructions and the lower 8-bits can be modified for computed addresses by file register instructions. (Note: The upper 2 bits are not affected.) In addition, an on-chip six level stack is employed to push and pull the contents of the program counter. This provides easy to use subroutine nesting. Activating the $\overline{MCLR}$ input on power-up initializes the external ROM program to address 1777_8.

PIN FUNCTIONS

Signal	Function
OSC 1 (Input) OSC 2 (Output)	Oscillator pins. The on-board oscillator can be driven by an external crystal, an RC network, or an external clock via these pins.
$\overline{RT}$ (Input)	Real Time input. Negative transitions on this pin increments the RTCC (F6) register. This pin can also be used for an interrupt input. This pin uses a Schmitt trigger input. There is no internal active pull-up device.
RA0-RA7, RB0-RB7, RC0-RC7, RD0-RD7	User programmable input/output lines. These lines can be inputs and/or outputs and are under direct control of the program.
$\overline{MCLR}$ (Input)	Master Clear. Used to initialize the internal ROM program to address 1777_8, latch all I/O registers high, and disable the interrupt system. This pin uses a Schmitt trigger input. There is no internal active pull-up device.
V_{DD}	Power supply pin.
V_{SS}	Ground pin.
CLKOUT	Clock Output. A signal derived from the internal oscillator. May be used by external circuitry to synchronize with PIC1665 timing.
HALT/ACK	Halt/Halt Acknowledge. This is a bidirectional I/O pin used in conjunction with CLKOUT. The pin is an input when CLKOUT is low, and an output when CLKOUT is high. Inputting a high when CLKOUT is low will suspend execution of the next instruction. No data is lost and after HALT/ACK is brought low execution proceeds exactly as if no halt signal had been applied. This pin can also be used to restart the PIC1665 after a HALT instruction (00001_8) has been executed. During the time CLKOUT is high, the pin will have an open drain output configuration and therefore requires an external pullup resistor. The output is high whenever the PIC1665 is halted either due to an active input to the HALT/ACK pin or the execution of the HALT instruction. HALT/ACK will output a low when the PIC1665 resumes execution of the program. This pin must be grounded when it is not used.
D0-D12 (Input)	Data input. These thirteen lines accept the thirteen bit PIC instruction codes generated by an external source. D0 is the LSB of the instruction.
A0-A10 (Output)	Address Output. These eleven lines represent the address of the next instruction to be executed by the PIC1665. They are capable of addressing up to 2048 words of memory. A0 is the LSB of the address.

PIN CONFIGURATION
64 LEAD DUAL IN LINE



REGISTER FILE ARRANGEMENT

File Octal	Function
F0	Not a physical register. F0 calls for the contents of the FSR (F4) to be used to select a file register. F4 is used as an indirect address pointer.
F1	W Register — The working register.
F2	Program Counter — Points to the next program ROM address to be executed.
F3	Arithmetic Status Register 7 6 5 4 3 2 1 0 0 X A9 A8 OV Z DC C Bit 0 (C) — Bit 0 is the carry flag which is usually the carry from the A.L.U. It is also used as a borrow in subtract instructions. Bit 1 (DC) — Bit 1 is the half carry (decimal carry) and is used to indicate a carry from bit 3 in the A.L.U. as the result of an addition (byte). This bit is used in the decimal adjust instruction to allow B.C.D. decimal addition. Bit 2 (Z) — Bit 2 is the zero flag and is set to a one if the results of the previous operation was identically zero. Bit 3 (OV) — Bit 3 is the overflow flag, and is set to a one by operations which cause a signed two's complement arithmetic overflow. The bit is set when the carry from the MSB in the A.L.U. is opposite to the carry from the MSB-1 bit. Bit 4 (A8) — Bit 4 is the 9th bit of the program counter. This bit is a read only bit. Bit 5 (A9) — Bit 5 is the 10th bit of the program counter. This bit is a read only bit.
F4	File Select Register — The FSR is used in generating effective file register addresses under program control.
F5	Interrupt Status Register — Used to control interrupts and F6 and F7. X CNTE A/B CNTS RTCIR XIR RTCIE XIE
F6,F7	RTCCA and RTCCB — Real Time Clock Counters A & B respectively can be arranged as two 8 bit registers, a single 16 bit register or two general purpose registers when no external counting is required. The RTCC registers can be loaded and read by the program, as well as count negative transitions on the RT pin or count at 1/8 the frequency of the oscillator. If data are being stored into RTCCA simultaneously with a negative transition on the RT pin (and CNTE = 1 and CNTS = 1), RTCCA will contain the new stored value and the external transition will be ignored by the microcomputer. (See the section "Real-Time Clock Interrupt" for further details about the RTCC registers.)
F10,11	I/O Port A
F12,13	I/O Port B
F14,15	I/O Port C
F16,17	I/O Port D
F20,77	General Purpose Registers—Used for temporary and general purpose storage during program execution time.

NOTE: F10, 12, 14 & 16 are the I/O registers and F11, 13, 15 & 17 are used for reading the actual pin levels.

MICROCOMPUTER

INTERRUPT SYSTEM

The interrupt system of the PIC1665 is comprised of an external interrupt and a real-time clock counter interrupt. These have different interrupt vectors, enable bits and status bits. Both interrupts are controlled by the status register (F5)** shown below.

NOT USED	CNTE	A/B	CNTS	RTCIR	XIR	RTCIE	XIE
7*	6	5	4	3	2	1	0

*Bit 7 is unused and is read as zero.

**Register 5 will power up to all zeroes.

EXTERNAL INTERRUPT

On any high to low transition on the $\overline{RT}$ pin the external interrupt request (XIR) bit will be set. This request will be serviced if the external interrupt enable (XIE) bit is set or if it is set at a later point in the program. The latter allows the processor to store a request (without interrupting) while a critical timing routine is being executed. Once external interrupt service is initiated, the processor will clear the XIR bit, delay one cycle (to execute the current instruction), then push the current program counter on to the stack and execute the instruction at location 1760_8. It takes three to four instruction cycles from the transition on the $\overline{RT}$ pin until the instruction at 1760_8 is executed. No new interrupts can be serviced until a return from interrupt (RETFI) instruction has been executed.

REAL-TIME CLOCK INTERRUPT

The real-time clock counter (RTCCA & RTCCB, file registers F6 and F7) have a similar mechanism of interrupt service. The RTCCA register will increment if the count enable (CNTE) bit is set. If this bit is not set the RTCCA & RTCCB will maintain their present contents and can therefore be used as general purpose RAM registers. The count source (CNTS) bit selects the clocking source for RTCCA. If CNTS is cleared to a '0', then RTCCA will use the internal instruction clock and increment at 1/8 the frequency present on the OSC pins. If CNTS is set to a '1', then RTCCA will increment on each high to low transition of the RT pin. RTCCB can only be incremented when RTCCA makes a transition from 377_8 to 0 and the A/B status bit is set. This condition links the two eight bit registers together to form one sixteen bit counter. An interrupt request under these conditions will occur when the combined registers make a transition from 177777_8 to 0. If, however, the A/B bit is not set, then RTCCA will be the only incrementing register and an interrupt request will occur when RTCCA makes a transition from 377_8 to 0. (In this setup the RTCCB register will not increment and can be used as a general purpose RAM register). Once a request has come from the real-time clock counter, the real-time clock interrupt request (RTCIR) bit will be set. At this point, the request can either be serviced immediately if the real-time clock interrupt enable (RTCIE) bit is set or be stored if RTCIE is not set. The latter allows the processor to store a real-time clock interrupt while a critical timing routine is being executed. Once interrupt service is initiated, the processor will clear the RTCIR bit, delay one cycle (to execute the current instruction), then push the present program counter on to the stack and execute the instruction at location 1740_8. It takes three instruction cycles from when the RTCC (A or B) overflows until the instruction at 1740_8 is executed. No new interrupts can be serviced until a RETFI instruction has been executed.

The RETFI instruction (00002_8) must be used to return from any interrupt service routine if any pending interrupts are to be serviced. External interrupts have priority over RTCC driven interrupt in the event both types occur simultaneously. Interrupts cannot be nested but will be serviced sequentially. The existence of any pending interrupts can be tested via the state of the XIR (bit 2) and RTCIR (bit 3) in the status word F5.

HALT OPERATION

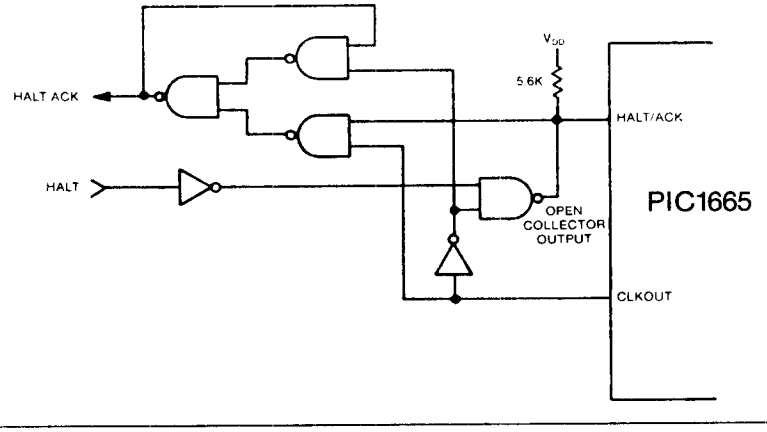
Program execution in the PIC1665 can be suspended in two ways. The first is by applying a logic high input level on the HALT/ACK pin when CLKOUT is low. The operation of the PIC1665 will be suspended until the HALT/ACK pin is brought low. At that point program execution will begin with the next instruction present on the Data Lines. Program execution can also be suspended using the HALT instruction. In order to restart the PIC1665 after execution of a HALT instruction, the $\overline{MCLR}$ pin must be brought low, or the HALT/ACK pin must be brought high for one complete cycle and then low again.

In both cases, when CLKOUT is high, the HALT/ACK pin will output a high level whenever the PIC1665 is in the halt mode provided an external pullup resistor is used.

When the PIC1665 is in the halt mode the RTCCA and RTCCB registers cannot be incremented by the internal clock, or by high to low transitions on the $\overline{RT}$ pin. If a high to low transition occurs on the $\overline{RT}$ pin, then the XIR bit (Bit 2 of file 5) will be set. If the XIE bit (bit 0 of file 5) is set, then an interrupt will occur immediately after program execution begins.

MICROCOMPUTER

TYPICAL HALT CIRCUIT



Basic Instruction Set Summary

Each PIC instruction is a 13-bit word divided into an OP code which specifies the instruction type and one or more operands which further specify the operation of the instruction. The following PIC instruction summary lists byte-oriented, bit-oriented, and literal and control operations.

For byte-oriented instructions, "f" represents a file register designator and "d" represents a destination designator. The file register designator specifies which one of the PIC file registers is to be utilized by the instruction. The destination designator specifies where the result of the operation performed by the instruction is to be placed. If "d" is zero, the result is placed in the PIC W register. If "d" is one, the result is returned to the file register specified in the instruction.

For bit-oriented instructions, "b" represents a bit field designator which selects the number of the bit affected by the operation, while "f" represents the number of the file in which the bit is located.

For literal and control operations, "k" represents an eight or nine bit constant or literal value.

For an oscillator frequency of 5MHz the instruction execution time is 2.0μsec, unless a conditional test is true or the program counter is changed as a result of an instruction. In these two cases, the instruction execution time is 4.0 μsec.

BYTE ORIENTED FILE REGISTER OPERATIONS

(12-7)	(6)	(5-0)
OP CODE	d	f (FILE #)

Instruction—Binary (Octal)					Name	Mnemonic,	Operands	Operation	Status Affected
0 0 0	0 0 0	0 0 0	1 0 0	(00004)	Decimal adjust W	DAW	—	(Note 1)	C
0 0 0	0 0 1	f f f	f f f	(00100)	Move W to file	MOVWF	f	W→f	—
0 0 0	1 d	f f f	f f f	(00200)	Subtract W from file w/borrow	SUBBWF	f,d	f+$\overline{W}$+c →d	OV,C,DC,Z
0 0 0	1 0 d	f f f	f f f	(00400)	Subtract W from file	SUBWF	f,d	f+$\overline{W}$ + 1 →d	OV,C,DC,Z
0 0 0	1 1 d	f f f	f f f	(00600)	Decrement file	DECF	f,d	f - 1 →d	OV,C,DC,Z
0 0 1	0 0 d	f f f	f f f	(01000)	Inclusive or W with file	IORWF	f,d	W∨f→d	Z
0 0 1	0 1 d	f f f	f f f	(01200)	And W with file	ANDWF	f,d	W•f→d	Z
0 0 1	1 0 d	f f f	f f f	(01400)	Exclusive OR W with file	XORWF	f,d	W⊕f→d	Z
0 0 1	1 1 d	f f f	f f f	(01600)	Add W with file	ADDWF	f,d	W+f→d	OV,C,DC,Z
0 1 0	0 0 d	f f f	f f f	(02000)	Add W to file with carry	ADCWF	f,d	W+f+c→d	OV,C,DC,Z
0 1 0	0 1 d	f f f	f f f	(02200)	Complement file	COMPF	f,d	$\overline{f}$→d	Z
0 1 0	1 0 d	f f f	f f f	(02400)	Increment file	INCF	f,d	f+1→d	OV,C,DC,Z
0 1 0	1 1 d	f f f	f f f	(02600)	Decrement file, skip if zero	DECFSZ	f,d	f - 1→ d, skip if zero	—
0 1 1	0 0 d	f f f	f f f	(03000)	Rotate file right thru carry	RRCF	f,d	f(n)→d(n−1), c→d(7), f(0)→c	C
0 1 1	0 1 d	f f f	f f f	(03200)	Rotate file left thru carry	RLCF	f,d	f(n)→d(n+1), c→d(0), f(7)→c	C
0 1 1	1 0 d	f f f	f f f	(03400)	Swap upper and lower nibble of file	SWAPF	f,d	f(0-3)⇌(4-7)→d	—
0 1 1	1 1 d	f f f	f f f	(03600)	Increment file, skip if zero	INCFSZ	f,d	f+1 →d, skip if zero	—

(12-6)	(5-0)
OP CODE	f (FILE #)

Instruction—Binary (Octal)					Name	Mnemonic,	Operands	Operation	Status Affected
1 0 0 0	0 0 0	f f f	f f f	(10000)	Move file to W	MOVFW	f	f →W	Z
1 0 0 0	0 0 1	f f f	f f f	(10100)	Clear file	CLRF	f	0 →f	Z
1 0 0 0	0 1 0	f f f	f f f	(10200)	Rotate file right/no carry	RRNCF	f	f(n)→d(n−1), f(0), →f(7)	—
1 0 0 0	0 1 1	f f f	f f f	(10300)	Rotate file left/no carry	RLNCF	f	f(n)→d(n+1), f(7), →f(0)	—
1 0 0 0	1 0 0	f f f	f f f	(10400)	Compare file to W, skip if F < W	CPFSLT	f	f - W, Skip if C = 0	—
1 0 0 0	1 0 1	f f f	f f f	(10500)	Compare file to W, skip if F = W	CPFSEQ	f	f - W, Skip if Z = 1	—
1 0 0 0	1 1 0	f f f	f f f	(10600)	Compare file to W, skip if F > W	CPFSGT	f	f - W, Skip if $\overline{Z}$ • C = 1	—
1 0 0 0	1 1 1	f f f	f f f	(10700)	Move file to itself	TESTF	—	f →f	Z

BIT ORIENTED FILE REGISTER OPERATIONS

(12-9)	(8-6)	(5-0)
OP CODE	b (BIT #)	f (FILE #)

Instruction—Binary (Octal)					Name	Mnemonic,	Operands	Operation	Status Affected
0 1 0 0	b b b	f f f	f f f	(04000)	Bit clear file	BCF	f,b	0→f(b)	—
0 1 0 1	b b b	f f f	f f f	(05000)	Bit set file	BSF	f,b	1→f(b)	—
0 1 1 0	b b b	f f f	f f f	(06000)	Bit test, skip if clear	BTFSC	f,b	Bit Test f(b), skip if clear	—
0 1 1 1	b b b	f f f	f f f	(07000)	Bit test, skip if set	BTFSS	f,b	Bit Test f(b), skip if set	—

LITERAL AND CONTROL OPERATIONS

(12-8)	(7-0)
OP CODE	k (LITERAL)

Instruction—Binary (Octal)					Name	Mnemonic,	Operands	Operation	Status Affected
0 0 0	0 0 0	0 0 0	0 0 0	(00000)	No Operation	NOP	—	—	—
0 0 0	0 0 0	0 0 0	0 0 1	(00001)	Halt in PIC1665	HALT	—	—	—
0 0 0	0 0 0	0 0 0	0 1 0	(00002)	Return from Interrupt	RETFI	—	Stack → PC	—
0 0 0	0 0 0	0 0 0	0 1 1	(00003)	Return from Subroutine	RETFS	—	Stack → PC	—
1 0 0	0 k k	k k k	k k k	(11000)	Move Literal to W	MOVLW	k	k→W	—
1 0 0	1 k k	k k k	k k k	(11400)	Add Literal to W	ADDLW	k	k+W→W	OV,C,DC,Z
1 0 1	0 k k	k k k	k k k	(12000)	Inclusive OR Literal to W	IORLW	k	k∨W→W	Z
1 0 1	0 1 k	k k k	k k k	(12400)	And Literal and W	ANDLW	k	k•W→W	Z
1 0 1	1 0 k	k k k	k k k	(13000)	Exclusive OR Literal and W	XORLW	k	k⊕W→W	Z
1 0 1	1 1 k	k k k	k k k	(13400)	Return and load literal in W	RETLW	k	k→W, Stack →PC	—

(12-10)	(9-0)
OP CODE	k (LITERAL)

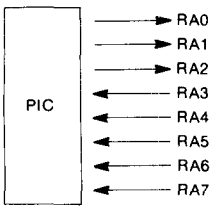
Instruction—Binary (Octal)					Name	Mnemonic,	Operands	Operation	Status Affected
1 0 k	k k k	k k k	k k k	(14000)	Go to address	GOTO	k	k→PC	—
1 1 k	k k k	k k k	k k k	(16000)	Call Subroutine	CALL	k	PC+1→Stack, k→PC	—

Note 1:
DAW: Decimal Adjust W
This instruction adjusts the eight bit number in the W register to form two valid BCD (binary coded decimal) digits,one in the lower and one in the upper nibble. (The results will only be meaningful if the number in W to be adjusted is the result of adding together two valid two digit BCD numbers.)

The adjustment obeys the following two step algorithm:
1. If the lower nibble is greater than 9 or the digit carry flag (DC) is set, 06 is added to the W register.
2. Then, if the upper nibble is greater than 9 or the carry from the original or step 1 addition is set, 60 is added to the W register. The carry bit is set if there is a carry from the original, step 1 or step 2 addition.

INPUT/OUTPUT CAPABILITY

The PIC1665 provides four complete quasi-bidirectional input/output ports. A simplified schematic of an I/O pin is shown below. The ports occupy address locations in the register file space of the PIC1665. Thus, any instruction that can operate on a general purpose register can operate on an I/O port. Two locations in the register file space are allocated for each I/O port. Port RA0-7 is addressable as either F10 or F11. Port RB0-7 is addressable as either F12 or F13. Port RC0-7 is addressable as either F14 or F15 and Port RD0-7 is addressable as either F16 or F17. An I/O port READ on its odd-numbered location will interrogate the chip pins while an I/O port READ on its even-numbered location will interrogate the internal latch in that I/O port. This simplifies programming in cases where a portion of a single port is used for inputting only, while the remainder is used for outputting as illustrated in the following example.



Here, the low 3 bits of port RA are used as output-only, while the high 5 bits are used as input-only. During power on reset ($\overline{MCLR}$ low), the latches in the I/O ports will be set high, turning off all pull down transistors as represented by Q_2 in Figure 1. During program execution if we wish to interrogate an input pin, then, for example,

```
BTFSS 11,6
```

will test pin RA6 and skip the next instruction if that pin is set. If we wish to modify a single output, then, for example,

```
BCF 10,2
```

will force RA2 to zero because its internal latch will be cleared to zero. This will turn on A_2 and pull the pin to zero.

The way this instruction operates internally is the CPU reads file 10 into the A.L.U., modifies the bit and re-outputs the data to file 10. If the pins were read instead, any input which was grounded externally would cause a zero to be read on that bit. When the CPU re-outputted the data to the file, that bit would be cleared to zero, no longer useful as an input until set high again.

During program execution, the latches in bits 3-7 should remain in the high state. This will keep A_2 off, allowing external circuitry full control of pins RA3-RA7, which are being used here as input.

BIDIRECTIONAL INPUT-OUTPUT PORT

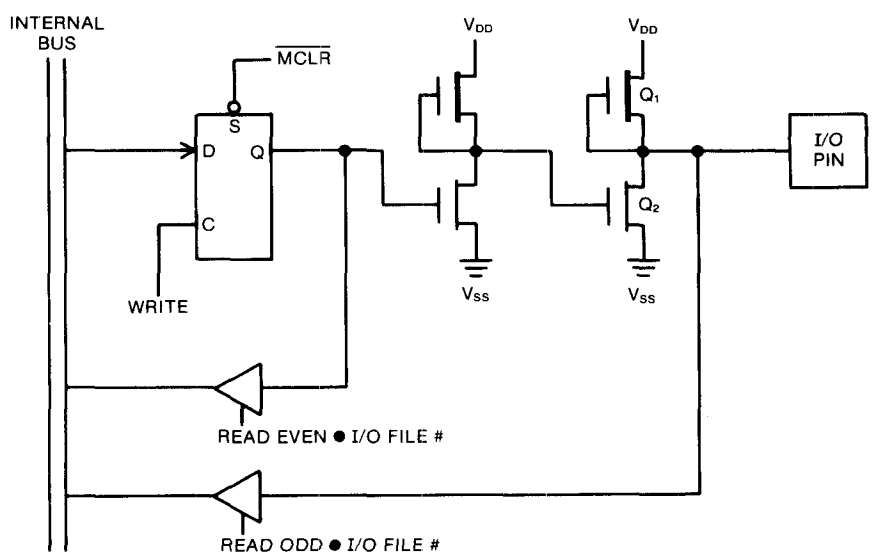


Figure 1

MICROCOMPUTER

ELECTRICAL CHARACTERISTICS

Maximum Ratings*

Ambient Temperature Under Bias ..... 70°C
Storage Temperature ..... −55°C to +150°C
Voltage on any Pin with Respect to V_{SS} ..... −0.3V to +10.0V
Power Dissipation 1000mW

*Exceeding these ratings could cause permanent damage to the device. This is a stress rating only and functional operation of this device at these conditions is not implied—operating ranges are specified in Standard Conditions. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Standard Conditions (unless otherwise stated):

DC CHARACTERISTICS

Operating Temperature $T_A = 0°C$ to $+70°C$

Data labeled "typical" is presented for design guidance only and is not guaranteed.

Characteristic	Sym	Min	Typ†	Max	Units	Conditions
Primary Supply Voltage	V_{DD}	4.5	—	5.5	V	
Primary Supply Current	I_{DD}	—	—	100	mA	All I/O pins high
Input Low Voltage (except $\overline{MCLR}$ & $\overline{RT}$)	V_{IL}	0.2	—	0.8	V	
Input High Voltage (except MCLR, RT, OSC1)	V_{IH1}	2.4	—	V_{DD}	V	
Input High Voltage ($\overline{MCLR}$, $\overline{RT}$, OSC1)	V_{IH2}	$V_{DD}-1$	—	V_{DD}	V	
Output High Voltage	V_{OH}	2.4	—	V_{DD}	V	$I_{OH} = -100\mu A$ provided by internal pullups (Note 2)
Output Low Voltage (I/O, A0-A9, HALT ACK, CLK OUT)	V_{OL}	—	—	0.45	V	$I_{OL} = 1.6mA$
Input Leakage Current ($\overline{MCLR}$, $\overline{RT}$,OSC1)	I_{LC}	−5	—	+5	μA	$V_{SS} \leq V_{IN} \leq V_{DD}$
Input Low Current (all I/O ports)	I_{IL}	0.2	0.6	−2.0	mA	$V_{IL} = 0.4V$, internal pullup
Input High Current (all I/O ports)	I_{IH}	0.1	0.4	—	mA	$V_{IH} = 2.4V$

† Typical data is at $T_A = 25°C$, $V_{DD} = 5.0V$.

NOTES:

1. Total power dissipation for the package is calculated as follows:
   $P_D = (V_{DD})(I_{DD}) + \Sigma (V_{DD} - V_{IL})(I_{IL}) + \Sigma (V_{DD} - V_{OH})(I_{OH}) + \Sigma (V_{OL})(I_{OL})$.
2. Positive current indicates current into pin. Negative current indicates current out of pin.
3. Total I_{OL} for all output pin (I/O ports plus CLK OUT) must not exceed 175mA.

Standard Conditions (unless otherwise stated):

AC CHARACTERISTICS

Operating Temperature $T_A = 0°C$ to $+70°C$

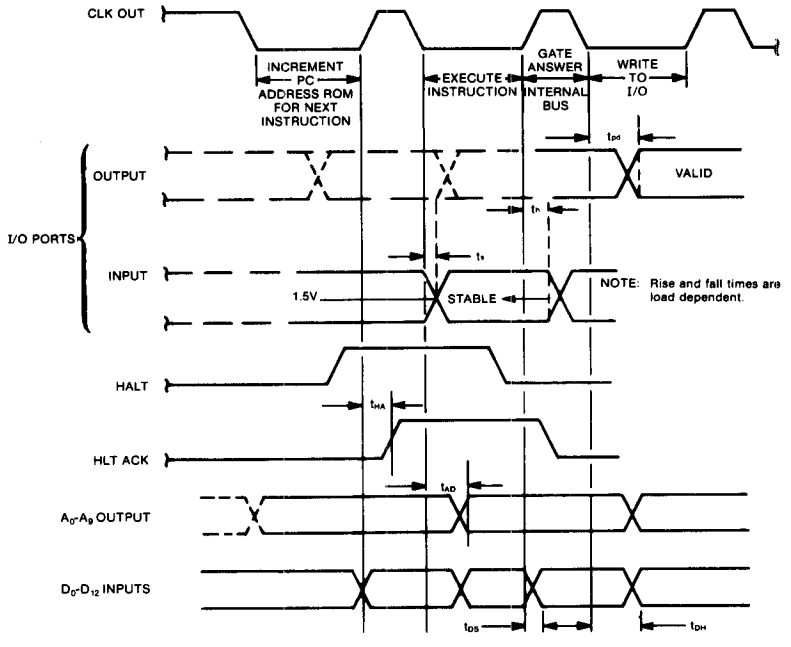
Characteristic	Sym	Min	Typ	Max	Units	Conditions
Instruction Cycle Time	t_{cy}	2.0	—	8	μs	4MHz—.1MHz external time base (Note 1)
$\overline{RT}$ Input						(Note 2)
Period	t_{RT}	$t_{CY}+0.2\mu s$	—	—	—	(Notes 2 and 3)
High Pulse Width	t_{RTH}	$\frac{1}{2}t_{RT}$	—	—	—	
Low Pulse Width	t_{RTL}	$\frac{1}{2}t_{RT}$	—	—	—	
I/O Ports						
Data Input Setup Time	t_s	—	—	$\frac{1}{4}t_{cy}-125$	ns	
Data Input Hold Time	t_h	0	—	—	ns	
Data Output Propagation Delay	t_{pd}	—	500	800	ns	Capacitive load = 50pF
HALT ACK Output Propagation Delay	t_{HA}	—	200	—	ns	
A0-A9 Output Propagation Delay	t_{AD}	—	350	—	ns	
D0-D12 Input Set-Up Time	t_{DS}	0	—	—	ns	
D0-D12 Input Hold Time	t_{DH}	200	—	—	ns	

NOTES:

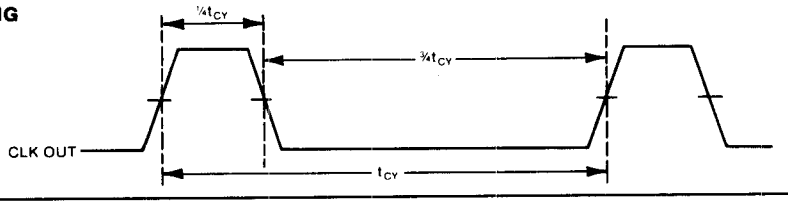
1. Instruction cycle period (t_{cy}) equals eight times the input oscillator time base period.
2. Due to the synchronous timing nature between CLK OUT and the sampling circuit used on the $\overline{RT}$ input, CLK OUT may be directly tied to the $\overline{RT}$ input. The minimum times specified represent theoretical limits.
3. The maximum frequency which may be input to the $\overline{RTCC}$ pin is calculated as follows: $f_{(max)} = \frac{1}{t_{RT\,(min)}} = \frac{1}{t_{CY\,(min)} + 0.2\mu s}$
   For example: if $t_{CY} = 4\mu s$, $f_{(max)} = \frac{1}{4.2\mu s} = 238KHz$.

MICROCOMPUTER

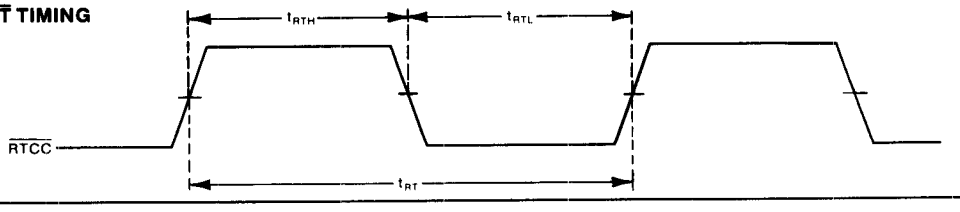
I/O TIMING



CLK OUT TIMING

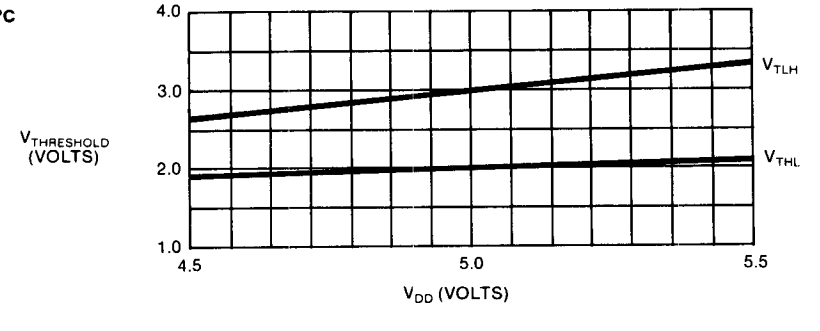


$\overline{RTCC}$/$\overline{RT}$ TIMING



SCHMITT TRIGGER CHARACTERISTICS (Typical)

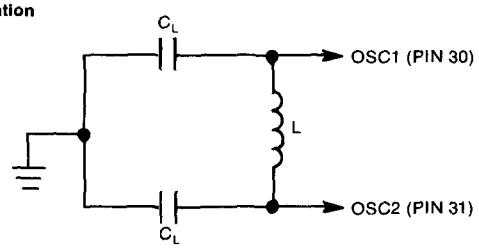
($\overline{RT}$, $\overline{MCLR}$) $T_A = 25°C$



MICROCOMPUTER

PIC1665 OSCILLATOR OPTIONS (Typical Circuits)

LC Operation

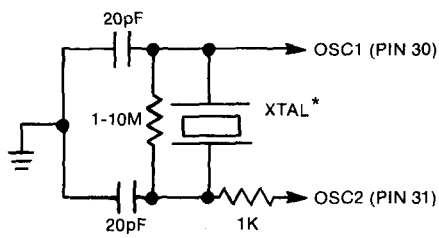


$$f_{OSC} \approx \frac{1}{2\pi\sqrt{L\,(C_L + C_{INT})}},$$

where $C_{INT} = 10pF$.

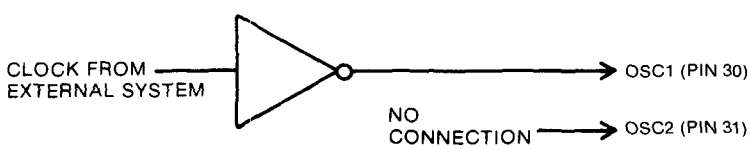
Typical values for 4MHz operation:
$L \approx 70\mu H$
$C_L = 10pF$

CRYSTAL INPUT OPERATION

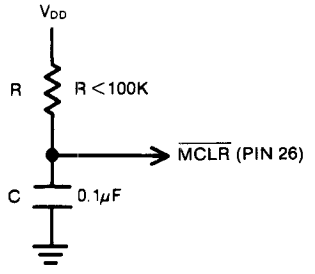


* or ceramic resonator, parallel resonant (0.8 - 5.0MHz).

EXTERNAL CLOCK INPUT OPERATION

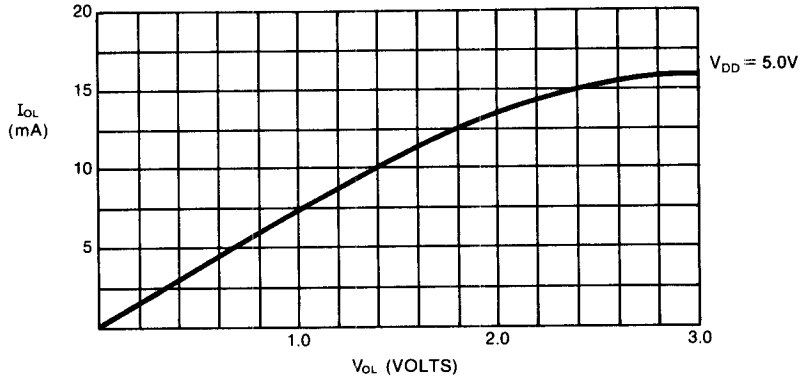


MASTER CLEAR (Typical Circuit)



Master Clear requires 10ms delay (assuming a 5MHz crystal) before activation after power is applied to the V_{DD} pin, for the crystal to start up. To achieve this, an external RC configuration as shown can be used (assuming V_{DD} is applied as a step function).

OUTPUT SINK CURRENT GRAPH



The Output Sink Current is dependent on the output load. This chart shows the typical curve used to express the output drive capability.

V_{OH} VS I_{OH} (I/O PORTS)