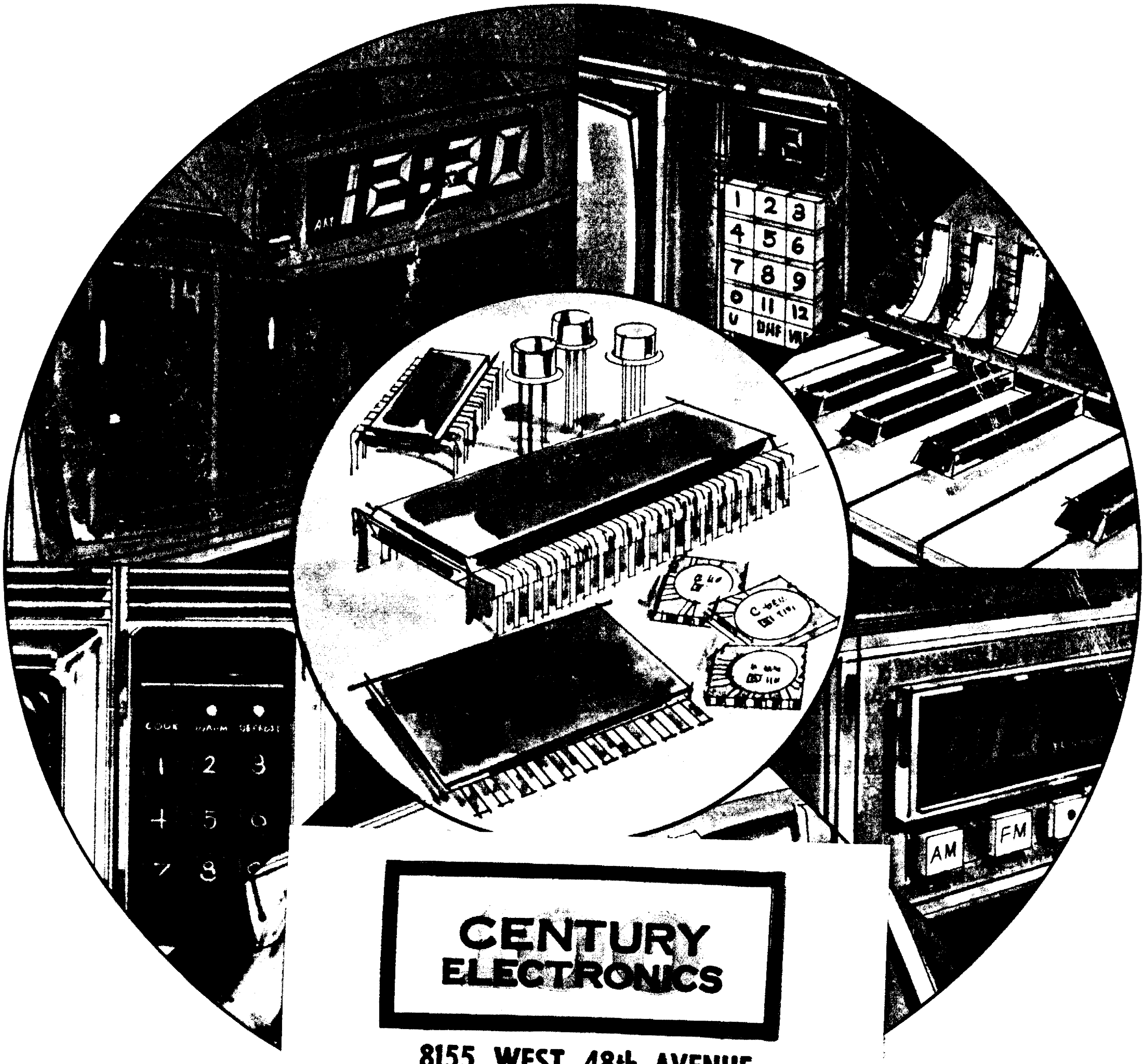


CENTURY ELECTRONICS

8155 WEST 48th AVENUE
WHEATRIDGE, COLORADO 80033

1977 DATA CATALOG

MICRO ELECTRONICS

GENERAL INSTRUMENT CORPORATION ▪ MICROELECTRONICS

1977 DATA CATALOG ▪ SECTIONS

PIC SERIES MICROCOMPUTER

FUNCTION	DESCRIPTION	PART NUMBER	PAGE NUMBER
8 BIT SINGLE CHIP MICRO-COMPUTER	The PIC 1650 MOS/LSI circuit array is a byte oriented programmable controller. The array is a complete chip controlled with an internal customer-defined ROM program specifying the overall functional characteristics and operational waveforms on each of the general purpose input/output lines.	PIC 1650	10C-2
8 BIT SINGLE CHIP DEVELOPMENT MICRO-COMPUTER	The PIC 1664 circuit is exactly the same as the PIC 1650 except that the ROM portion of the PIC 1650 has been removed. Any external RAM or PROM can be used to aid in the development of a final PIC 1650 configuration.	PIC 1664	10C-10

PIC SERIES MICROCOMPUTER

MICRO ELECTRONICS

PRELIMINARY INFORMATION

Programmable Intelligent Computer

FEATURES

- User Programmable
- Intelligent Controller for Stand-Alone Applications
- 32 8-Bit Registers
- 512 × 12-Bit ROM for Program
- Arithmetic Logic Unit
- 4 Sets of 8 User Defined TTL-compatible Input/Output Lines.
- Real Time Clock Counter
- Self contained Oscillator
- Access to RAM Registers inherent in instruction.

DESCRIPTION

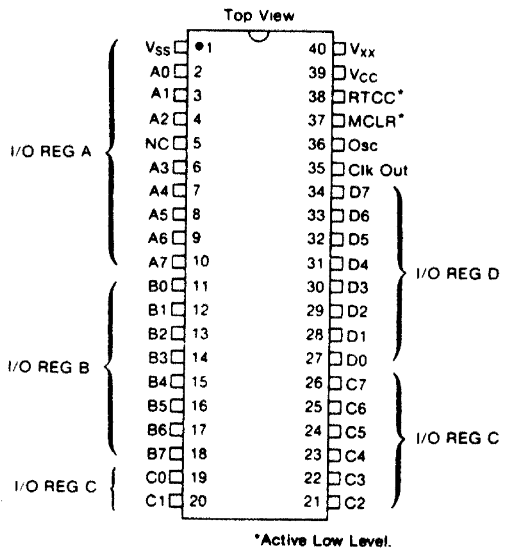
The PIC1650 MOS/LSI circuit array is a byte oriented programmable controller designed to satisfy the requirements for a low-cost, stand-alone 8-bit micro-computer. The array is a complete chip controlled with an internal customer-defined ROM program specifying the overall functional characteristics and operational waveforms on each of the general purpose input/output lines. The array can be programmed to scan keyboards, drive multiplexed displays, control vending machines, control traffic lights, control printers and to control automatic gasoline pumps. Since it contains ROM, RAM, I/O as well as the central processing unit on one device, the PIC1650 is truly a complete 8-bit micro-computer on one chip.

The PIC1650 is fabricated with N-Channel Ion Implant technology resulting in a high performance product with proven reliability and Production history. Only a single +5 volt power supply is required for operation, and an on-chip oscillator provides the operating clock with only an external R/C network to establish the frequency. Inputs and outputs are TTL compatible. The PIC1650 is supplied in a 40-pin dual-in-line package.

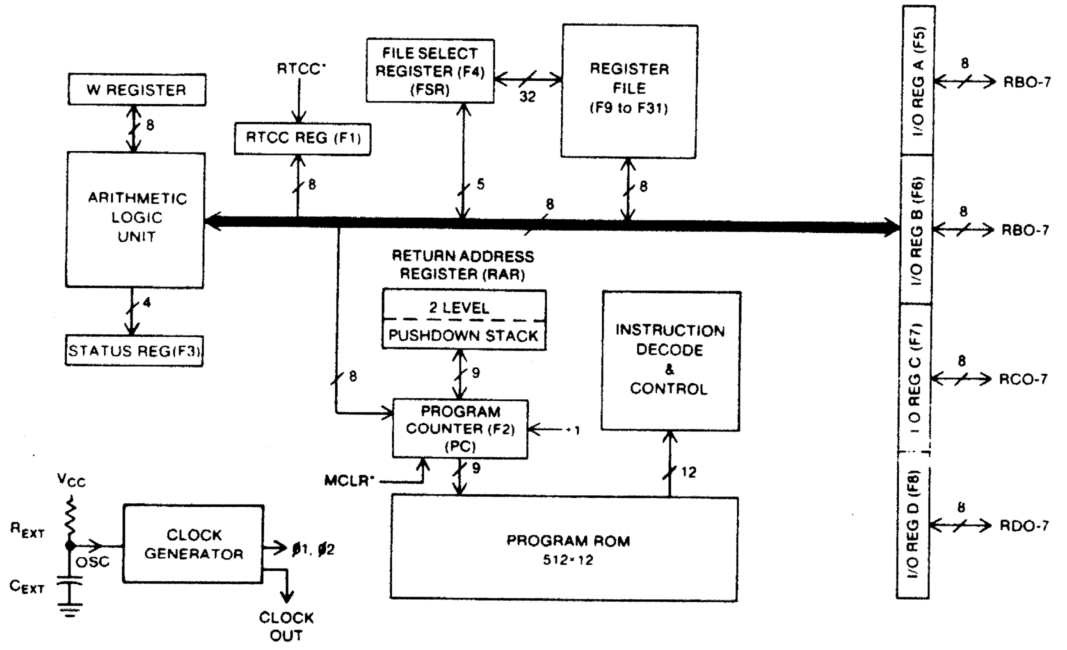
The PIC product family (PIC1650 and all extensions) is supported by an extensive software and hardware package. The software package includes Cross Assembler/Simulator programs designed to run on the large machine, on time share and minicomputer system levels. The hardware package includes a prototype TTL Emulator Board with which the user can verify, in his actual system, the program in either RAM or PROM before committing it to mask tooling. For added flexibility, the board can be interfaced into the GIMINI developmental system for conversational capability via a terminal with the PIC TTL Emulator's RAM memory. The PIC program is stored in RAM on the PIC Emulator board as memory as part of the CP1600 microprocessor address space. Thus, on-line changes in code can be implemented without the inconvenience of reburning PROMs.

PIN CONFIGURATION

40 LEAD DUAL IN LINE



BLOCK DIAGRAM



PIN FUNCTIONS

Signal	Function
OSC (Input)	Oscillator input. This signal can be driven by an external oscillator if a precise frequency of operation is required or an external R/C network can be used to set the frequency of operation of the internal clock generator. The maximum oscillator frequency is 1MHz.
RTCC* (Input)	Real Time Clock Counter. Used by the microprogram to keep track of elapsed time between events. The maximum RTCC* frequency is 250KHz. This register can be loaded and read by the program.
RA0-7, RB0-7, RC0-7, RD0-7 (Input/output)	User programmable input/output lines. These lines can be inputs and/or outputs and are under direct control of the program.
MCLR* (Input)	**Master Clear. Used to ihitialize the internal ROM program to address 777_8. Should be held low at least 20 μs past the time when the power supply is valid.**
CLOCK (output)	A signal derived from the internal oscillator. Used by external devices to synchronize themselves to PIC timing.

ARCHITECTURAL DESCRIPTION

The firmware architecture of the PIC1650 microcomputer is based on a register file concept with very simple, low level, commands designed to emphasize bit, byte, and register transfer operations. The primary purpose of the PIC is to perform logical processing, basic code conversions, formatting, and to generate fundamental timing and control signals for I/O devices. The instruction set also supports computing functions as well as these control and interface functions.

Internally, the PIC1650 is composed of three functional elements connected together by a single bidirectional bus: the Register File composed of 32 addressable 8-bit registers, an Arithmetic Logic Unit, and a Control ROM composed of 512 program words each 12 bits in width.

The Register File is divided into two functional groups: operational registers and general registers. The operational registers are addressed as F0 to F8 (the first 9 of the total of 32 file registers) and include, among others, the Real Time Clock Counter Register, the Status Register, the Program Counter (PC), and I/O Registers A, B, C and D (RA, RB, RC and RD). The general registers are addressed as F9 to F31 and are used for data and control information under command of the instructions.

The Logic Unit contains one temporary working register or accumulator (W Register) and gating to perform Boolean functions between data held in the working register and any file register.

The Control ROM contains the operational program for the rest of the logic within the controller. Sequencing of a micro-instructions is controlled via the Program Counter (PC) which automatically increments to execute in-line programs. Program control operations can be performed by Bit Test and Skip instructions, Jump instructions, or loading computed addresses into the PC. In addition, an on-chip pushdown stack is employed with the return address register serving as the top element of the stack. This permits easy to use subroutine nesting. Application of the +5V power supply initializes the ROM microprogram to address 777_8.

REGISTER FILE ARRANGEMENT

File	Function
F0	Not a physically implemented register. F0 calls for the contents of the File Select Register (low order 5 bits) to be used to select a file register. F0 is thus useful as an indirect address pointer. For example, W+F0 $\rightarrow$ W will add the contents of the file register pointed to by the FSR (F4) to W and place the result in W.
F1	Real Time Clock Counter Register. This register can be loaded and read by the microprogram. Clock keeps counting up after zero is reached.
F2	Program Counter (PC). The PC is automatically incremented and can be written into; e.g., MOVWF F2. If cannot be read, however.
F3	Status Word Register. The bits in this register can be set or cleared only by the Bit Set and Bit Clear instructions; they cannot be altered by other commands operating on F3.

(4)	(1)	(1)	(1)	(1)	
1	PC9	Z	DC	C	:F3

PC9:	Tenth bit of PC for future use. Future ROM space of 512-1023 can be addressed with this bit set.
C (Carry):	Stores the carry out on arithmetic operations, and acts as a bit link on rotate operations. This bit is set high on an SUBWF instruction if the addition of f, the one's complement of W, and a 1 results in a carry.
DC (Digit Carry):	Stores the carry out of low order digit on arithmetic operation. This bit is set high on a SUBWF instruction if the addition of f, the one's complement of W, and a 1 results in a carry from the low order digit.
Z (Zero):	Set if the result of the arithmetic operations is zero.

File	Function
F4	File Select Register (FSR). Low order 5 bits only are used. The FSR is used in generating effective file register addresses under program control. When accessed as a directly addressed file, the upper 3 bits read as a logic "1".
F5	I/O Register A (RA)
F6	I/O Register B (RB)
F7	I/O Register C (RC)
F8	I/O Register D (RD)

Instruction Set Summary

For an oscillator frequency of 1MHz, the instruction execution time is 4 μsec, except if a conditional test is true or if the PC register is changed as a result of an instruction. In these two cases, the instruction execution time is 8 μsec.

In the following PIC instruction descriptions "k" represents an eight bit constant or literal value, "f" represents a file register designator and "d" represents a destination designator. The file register designator specifies which one of the 32 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. If the "d" operand is omitted, the f register is assumed as the destination. "f" and "d" may be numbers, characters, or symbols as described in the PIC Assembler and PIC Simulator instructions. "C" represents the carry bit, "Z" represents the zero bit, and "DC" represents the digit carry bit.

GENERAL FILE REGISTER OPERATIONS

(6)	(1)	(5)
OP CODE	d	f (FILE #)

for d = 0, f→W
d = 1, f→f

Instruction (Octal)				Name	Syntax		Operation	Status
000000	0	00000	(0000)	No Operation	NOP	—	—	—
000000	1	fffff	(0040)	Move W to f *	MOVWF	f	W→f	—
000001	0	fffff	(0100)	Clear W	CLRW	—	O→W	Z
000001	1	fffff	(0140)	Clear f	CLRF	f	O→f	Z
000010	d	fffff	(0200)	Subtract W from f	SUBWF	f, d	f - W→d	C,DC,Z
000011	d	fffff	(0300)	Decrement f	DECF	f, d	f - 1→d	Z
000100	d	fffff	(0400)	Inclusive OR W and f	IORWF	f, d	W∨f→d	Z
000101	d	fffff	(0500)	AND W and f	ANDWF	f, d	W∧f→d	Z
000110	d	fffff	(0600)	Exclusive OR W and f	XORWF	f, d	W⊻f→d	Z
000111	d	fffff	(0700)	Add W and f	ADDWF	f, d	W+f→d	C,DC,Z
001000	d	fffff	(1000)	Move f	MOVF	f, d	f→d	Z
001001	d	fffff	(1100)	Complement f	COMF	f, d	$\bar{f}$→d	Z
001010	d	fffff	(1200)	Increment f	INCF	f, d	f+1→d	Z
001011	d	fffff	(1300)	Decrement f, Skip if Zero	DECFSZ	f, d	f - 1→d, skip if Zero	—
001100	d	fffff	(1400)	Rotate Right f	RRF	f, d	f(n)→d(n-1), f(0)→C, C→d(7)	C
001101	d	fffff	(1500)	Rotate Left f	RLF	f, d	f(n)→d(n+1), f(7)→C, C→d(0)	C
001110	d	fffff	(1600)	Swap halves f	SWAPF	f, d	f(0-3)⇄f(4-7)→d	—
001111	d	fffff	(1700)	Increment f, Skip if Zero	INCFSZ	f, d	f+1→d, skip if zero	—

BIT LEVEL FILE REGISTER OPERATIONS

(4)	(3)	(5)
OP CODE	b (BIT #)	f (FILE #)

Instruction (Octal)				Name	Syntax		Operation	Status
0100	bbb	fffff	(2000)	Bit Clear f	BCF	f, b	0→f(b)	—
0101	bbb	fffff	(2400)	Bit Set f	BSF	f, b	1→f(b)	—
0110	bbb	fffff	(3000)	Bit Test f, Skip if Clear	BTFSC	f, b	Bit Test f(b): skip if clear	—
0111	bbb	fffff	(3400)	Bit Test f, Skip if Set	BTFSS	f, b	Bit Test f(b): skip if set	—

LITERAL AND CONTROL OPERATIONS

(4)	(8)
OP CODE	l (LITERAL)

Instruction (Octal)			Name	Syntax		Operation	Status
1000	k k k k k k k k	(4000)	Return	RET	—	0 → W, RAR → PC	—
1000	k k k k k k k k	(4000)	Return and place Literal in W	RETLW	k	k → W, RAR → PC	—
1001	k k k k k k k k	(4400)	Call subroutine *	CALL	k	PC → RAR, k → PC	—
101x	k k k k k k k k	(5X00)**	Go To address	GOTO	k	k → PC	—
1100	k k k k k k k k	(6000)	Move Literal to W	MOVLW	k	k→W	—
1101	k k k k k k k k	(6400)	Inclusive OR Literal and W	IORLW	k	k∨W→W	Z
1110	k k k k k k k k	(7000)	AND Literal and W	ANDLW	k	k∧W→W	Z
1111	k k k k k k k k	(7400)	Exclusive OR Literal and W	XORLW	k	k⊻W→W	Z

*The 9th bit of the program counter in the PIC1650 is zero for a CALL and a MOVWF F2. Therefore, subroutines must be located in page 0. However, subroutines can be called from page 0 or page 1 since the RAR is 9 bits wide. (Page 0: 0-255, Page 1: 256-511)

OTHER INSTRUCTION MNEMONICS REGONIZED BY THE PIC1650 ASSEMBLER

Instruction (Octal)		Name	Syntax	Equivalent Operation(s)	Status
0100 000 00011	(2003)	Clear Carry	CLRC	BCF3, 0	—
0101 000 00011	(2403)	Set Carry	SETC	BSF3,0	—
0100 001 00011	(2043)	Clear Digit Carry	CLRDC	BCF3,1	—
0101 001 00011	(2443)	Set Digit Carry	SETDC	BSF3,1	—
0100 010 00011	(2103)	Clear Zero	CLRZ	BCF3, 2	—
0101 010 00011	(2503)	Set Zero	SETZ	BSF3,2	—
0111 000 00011	(3403)	Skip on Carry	SKPC	BTFSS3,0	—
0110 000 00011	(3003)	Skip on No Carry	SKPNC	BTFSC3,0	—
0111 001 00011	(3443)	Skip on Digit Carry	SKPDC	BTFSS3,1	—
0110 001 00011	(3043)	Skip on No Digit Carry	SKPNDC	BTFSC3, 1	—
0111 010 00011	(3503)	Skip on Zero	SKPZ	BTFSS3, 2	—
0110 010 00011	(3103)	Skip on No Zero	SKPNZ	BTFSC3, 2	—
001000 1 f f f f f	(1040)	Test File	TSTF f	MOVF f, 1	Z
001000 0 f f f f f	(1000)	Move File to W	MOVFW f	MOVF f, 0	Z
001001 1 f f f f f 001010 d f f f f f	(1140) (1200)	Negate File	NEGF, f,d	COMF f, 1 INCF f, d	Z
011000 0 00011 001010 d f f f f f	(3003) (1200)	Add Carry to File	ADDCF f, d	BTFSC 3,0 INCF f, d	Z
011000 0 00011 000011 d f f f f f	(3003) (0300)	Subtract Carry from File	SUBCF f,d	BTFSC 3,0 DECF f, d	Z
011000 1 00011 001010 d f f f f f	(3043) (1200)	Add Digit Carry to File	ADDDCF f,d	BTFSG 3,1 INCF f,d	Z
011000 1 00011 000011 d f f f f f	(3043) (0300)	Subtract Digit Carry from File	SUBDCF f,d	BTFSC 3,1 DECF f,d	Z
101× kkkkkkkk	(5×00)	Branch	B K	GO TO K	—
0110 00000011 101× kkkkkkkk	(3003) (5×00)	Branch on Carry	BC K	BTFSC 3,0 GO TO K	—
0111 00000011 101× kkkkkkkk	(3403) (5×00)	Branch on No Carry	BNC K	BTFSS 3,0 GO TO K	—
0110 00100011 101× kkkkkkkk	(3043) (5×00)	Branch on Digit Carry	BDC K	BTFSG 3,1 GO TO K	—
0111 00100011 101× kkkkkkkk	(3443) (5×00)	Branch on No Digit Carry	BNDC K	BTFSS 3,1 GO TO K	—
0110 01000011 101× kkkkkkkk	(3103) (5×00)	Branch on Zero	BZ K	BTFSC 3,2 GO TO K	—
0111 01000011 101× kkkkkkkk	(3503) (5×00)	Branch on No Zero	BNZ K	BTFSS 3,2 GO TO K	—

if × = 0, address is in page 0.
if × = 1, address is in page 1.

Sample Programs

In the following program steps, .dddd means literals, in decimal form, otherwise they are in octal form.

I OBJECTIVE: To display the file pointed to by F31 via I/O post A (F5). Assume .20 in F31 and .100 in F20.

Hence, the following program will display .100 via F5.

Program steps	Description
MOVF .31, W	Move the contents of F31 to the working register W. After execution W contains .20.
MOVWF 4	Move the contents of W to FSR (F4). After execution F4 contains .20.
MOVF 0, W	Move the contents of the file pointed to by FSR, that is, the contents of F20, to W. Thus W contains .100 after execution.
MOVWF 5	Move the contents of W to F5. Hence .100 is in F5 now.

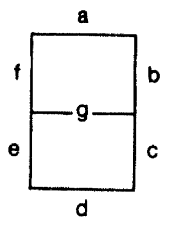
II OBJECTIVE: To compare F31 to a constant, if equal GOTO OK, if not equal GOTO NO.

Program Steps	Description
MOVF .31, W	Move the contents of F31 to the working register W
XORLW CONST	Exclusive Oring, bit by bit, the contents of W and the literal CONSTANT. If the are equal, all zero bits will result in W and the second bit in the status register (F3) will be set.
BTFSS 3,2	If the second bit in F3 is one, skip the next step.
GOTO NO	They are not equal.
GOTO OK	They are equal.

III OBJECTIVE: To clear files F5 to F31.

Program Steps	Description
MOVLW 4	Move the literal 4 to the working register W.
MOVWF 4	Move the literal 4 from W to the FSR (F4). These two steps initialize the pointer F4 to 4.
LPI INCF 4,5	Increment the contents of FSR by one. This is the same as saying that the pointer points to next file.
CLRF 0,F	Clear the contents of the file pointed to by FSR.
MOVLW .255	To sense the end of the file, which is F31, move the literal .255 to working register W. The literal .255 is used since the top three bits of F4 are read as one.
XORWF 4, W	Compare bit by bit the contents of FSR and W. If they are equal, the second bit in the status register (F3) will be set.
BTFSS 3,2	If they are not equal, go back to the step labelled LPI, otherwise stop.
GOTO LPI	
END	

IV OBJECTIVE: BCD to 7-Segment Code Conversion



Digit is displayed in either I/O Register 5, 6, 7, or 8. F20 contains the BCD numbers.

Program steps **Description**

```
          MOVLW   TBLSTR        ;   Starting address of table
          ADDWF   20, W         ;   Add BCD number as offset to Tablestart
          CALL    CONVRT        ;   Obtain converted code
          MOVWF   5             ;   Move the 7-segment code to the file pointed to by the file select register
            —
            —
            —
            —
            —
            —
            —
            —
            —
            —
            —
            —
CONVRT    MOVWF   2             ;   Move the computed address into the PC
TBLSTR    RETLW   B'1 1 1 1 1 1 0'  ;   0 in seven segment (abcdefg)
          RETLW   B'0 1 1 0 0 0 0'  ;   1 in seven segment (abcdefg)
          RETLW   B'1 1 0 1 1 0 1'  ;   2 in seven segment (abcdefg)
          RETLW   B'1 1 1 1 0 0 1'  ;   3 in seven segment (abcdefg)
          RETLW   B'0 1 1 0 0 1 1'  ;   4 in seven segment (abcdefg)
          RETLW   B'1 0 1 1 0 1 1'  ;   5 in seven segment (abcdefg)
          RETLW   B'0 0 1 1 1 1 1'  ;   6 in seven segment (abcdefg)
          RETLW   B'1 1 1 0 0 0 0'  ;   7 in seven segment (abcdefg)
          RETLW   B'1 1 1 1 1 1 1'  ;   8 in seven segment (abcdefg)
          RETLW   B'1 1 1 0 0 1 1'  ;   9 in seven segment (abcdefg)
```

Electrical Characteristics

Operating Parameters

V_{CC} Voltage = +5 volts ±5%

V_{XX} Voltage = +4.75V to +10.0V

I_{CC} Current = 50 ma typical; 75 ma max.

Temperature = 0°C to+50°C

Storage Temperature Range: -55°C to +150°C

D.C. Parameters

Input Logic "1"	V_{IH} = 2.4V min.
Input Logic "0"	V_{IL} = 0.8V max.
Input Leakage	I_{IL} = 15 μA max.
Input Capacitance	C_I = 5 pF max.
Output Logic "1"	V_{OH} = 2.4V min. @ 100 μA
Output Logic "0"	V_{OL} = 0.4V max. @ 1.8 mA
Output Leakage	I_{OL} = ± 5 μA max.
Output Capacitance	C_O = 10 pF max.

A.C. Parameters

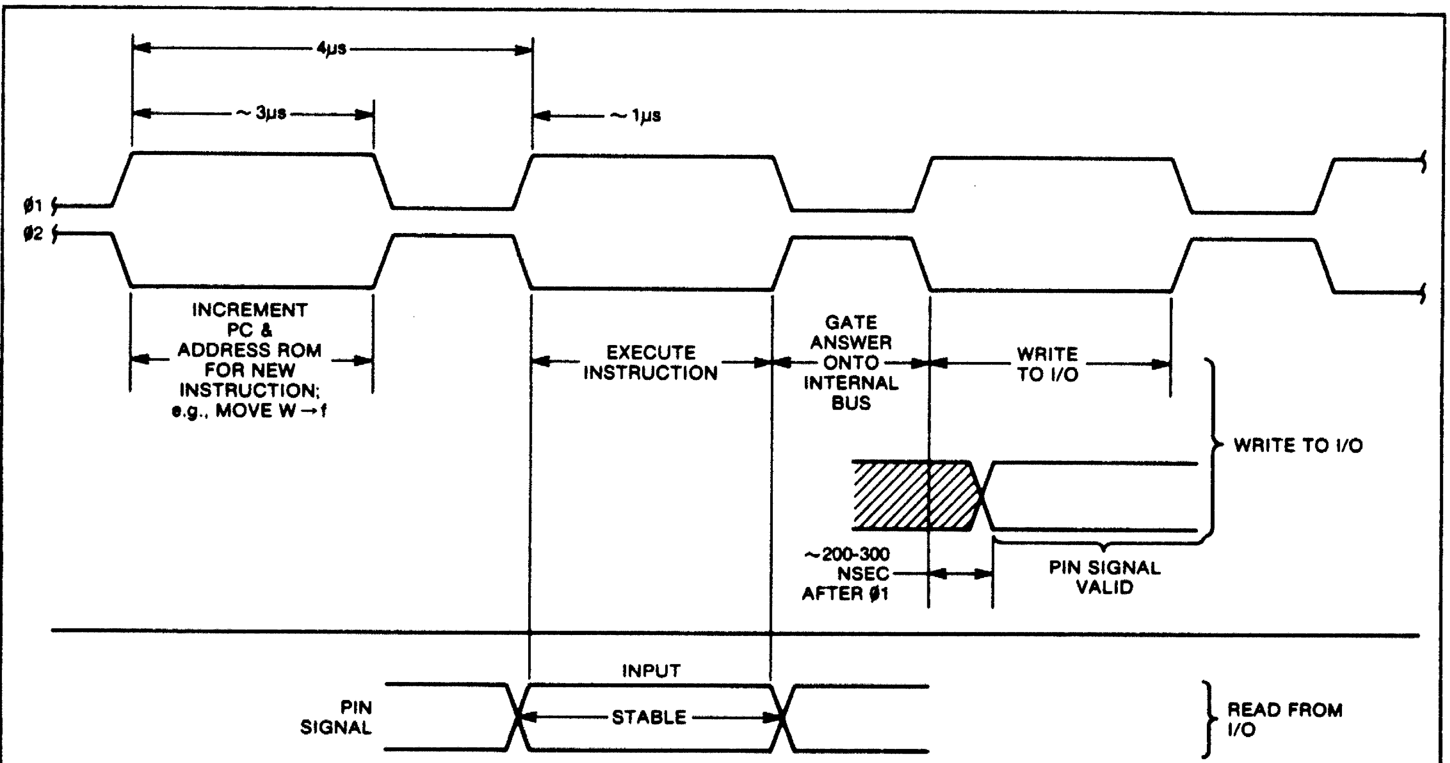
OSC Frequency	1 MHz
RTCC* Frequency	250 KC

LED Direct Drive

V_{XX} drives the gate of the output buffer, allowing adjustment of LED drive capability:

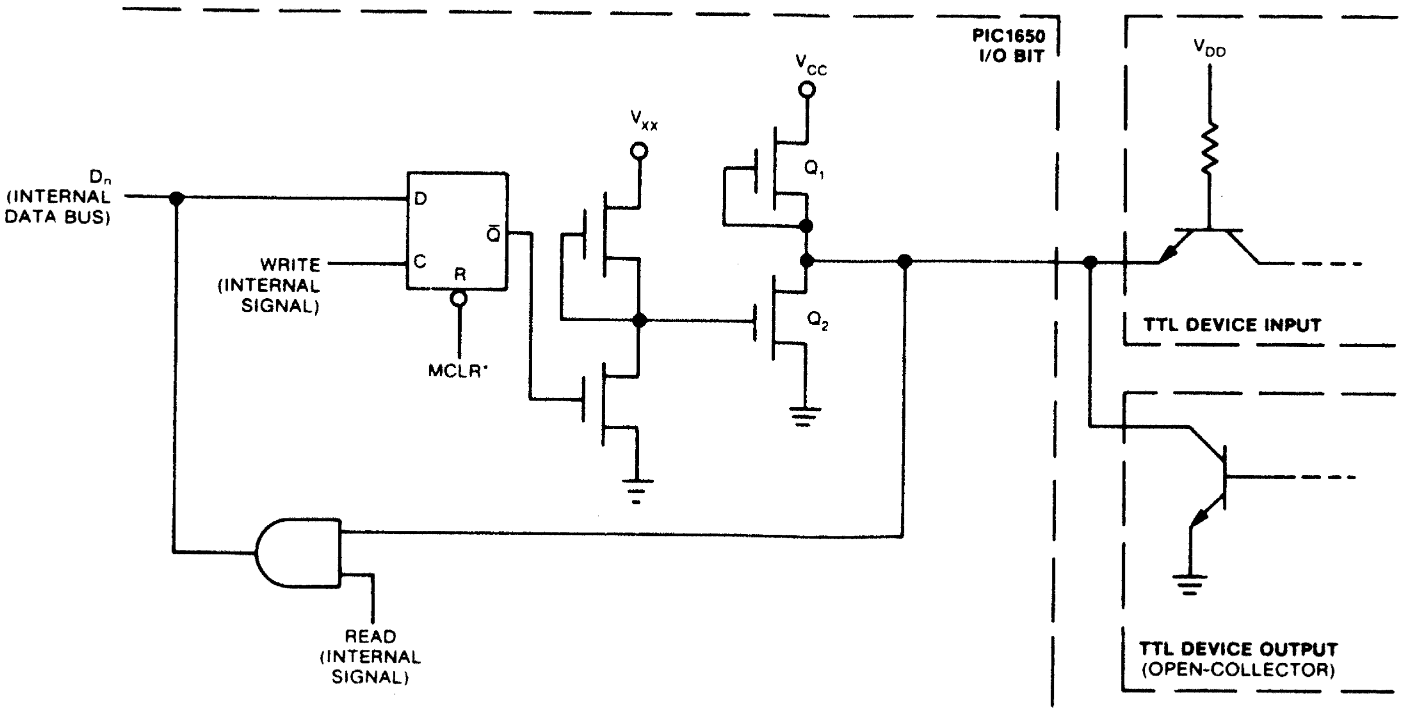
V_{XX}	V_{OUT}	I_{SINK} (typ.)
5V	0.4V	2.5mA
5V	0.7V	4.2mA
10V	0.4V	5.8mA
10V	0.7V	10.0mA
10V	1.0V	14.1mA

I/O Timing



I/O Interfacing

The equivalent circuit for an I/O port bit is shown below as it would interface with either the input of a TTL device (CPU chip is outputting) or the output of an open collector TTL device (CPU chip is inputting). Each I/O port bit can be individually time multiplexed between input and output functions under software control. When outputting thru a PIC I/O Port, the data is latched at the port and the pin can be connected directly to a TTL gate input. When inputting data thru an I/O Port, the port latch must first be set to a logic "1" level under program control. This turns off Q_2, allowing the TTL open collector device to drive the pad, pulled up by Q_1, which can source about 100μA.



I/O STRUCTURE

PRELIMINARY INFORMATION

Programmable Intelligent Computer Development Circuit

FEATURES

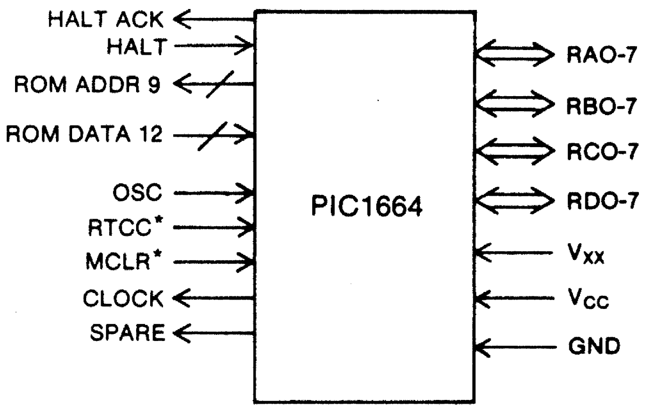
- PIC1650 computer with ROM removed
- Useful for engineering prototyping and field trial demonstrations
- PIC1650's ROM address & data lines brought out to pins
- PIC1650's ROM can be replaced by external RAM or PROM
- PIC1650 can be single stepped or stopped via the halt pin.

DESCRIPTION

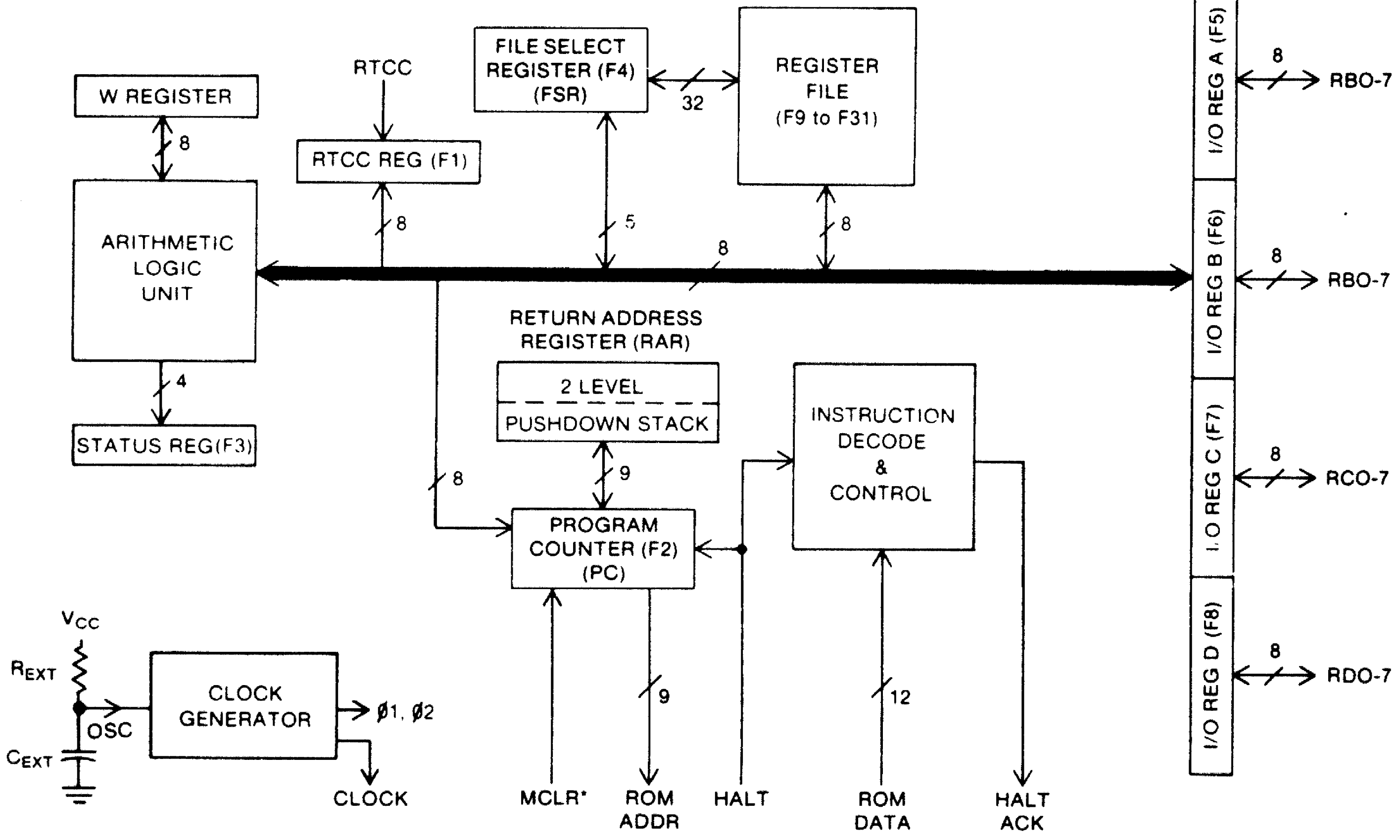
The PIC1664 MOS/LSI circuit array is exactly the same as PIC1650 except for the fact that the ROM is removed and that the ROM address and data lines are brought out, resulting in a 64 pin package. The addition of a halt pin has also been made to the 64 pin package. This pin gives the user the ability to stop as well as single-step the chip. The PIC1664 is designed as a useful tool for engineering prototyping and field trail demonstration.

PIN CONFIGURATION

64 LEAD DUAL IN LINE



BLOCK DIAGRAM



GENERAL INSTRUMENT CORPORATION
MICROELECTRONICS