National Semiconductor

PRELIMINARY

HPC16400/HPC36400/HPC46400 High-Performance Communications microController

General Description

The HPC16400 is a member of the HPC™ family of High Performance microControllers. Each member of the family has the same identical core CPU with a unique memory and I/O configuration to suit specific applications. Each part is fabricated in National's advanced microCMOS technology. This process combined with an advanced architecture provides fast, flexible I/O control, efficient data manipulation, and high speed computation.

The HPC16400 has 4 functional blocks to support a wide range of communication application-2 HDLC channels, 4 channel DMA controller to facilitate data flow for the HDLC channels, programmable serial interface and UART.

The serial interface decoder allows the 2 HDLC channels to be used with devices using interchip serial link for point-to-point & multipoint data exchanges. The decoder generates enable signals for the HDLC channels allowing multiplexed D and B channel data to be accessed.

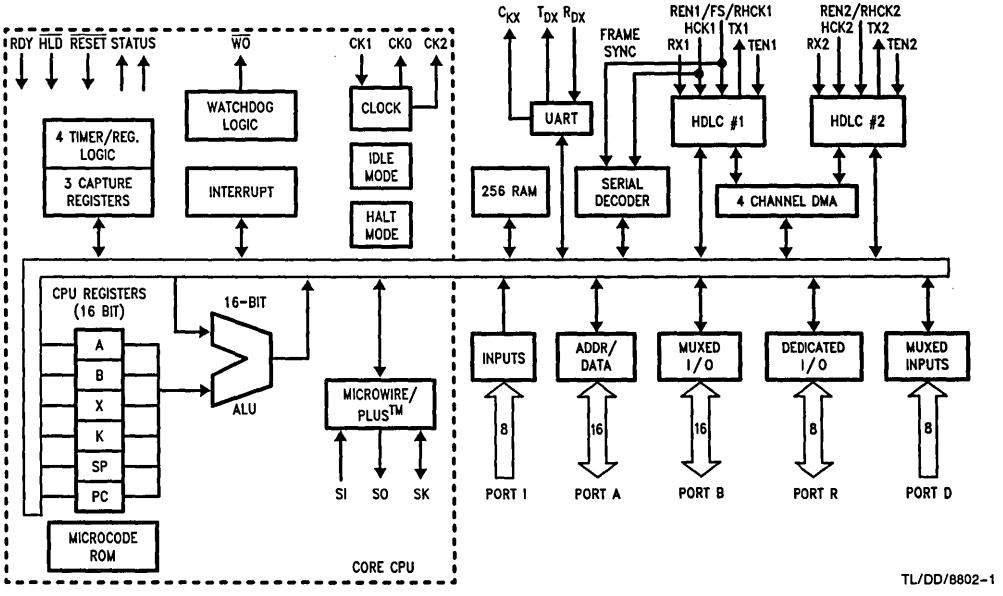
The HDLC channels manage the link by providing sequencing using the HDLC framing along with error control based upon a cyclic redundancy check (CRC). Multiple address recognition modes, and both bit and byte modes of operation are supported.

The HPC16400 is available in 68-pin PLCC, LCC, LDCC and 84-pin TapePak® packages.

Features

- HPC family—core features:
  - 16-bit data bus, ALU, and registers
  - 64 kbytes of external direct memory addressing
  - FAST!—20.0 MHz system clock
  - High code efficiency
  - 16 x 16 multiply and 32 x 16 divide
  - Eight vectored interrupt sources
  - Four 16-bit timer/counters with WATCHDOG logic
  - MICROWIRE/PLUS serial I/O interface
  - CMOS—low power with two power save modes
- Two full duplex HDLC channels
  - Optimized for X.25 and LAPD applications
  - Programmable frame address recognition
  - Up to 4.65 Mbps serial data rate
  - Built in diagnostics
  - Synchronous bypass mode
- Programmable interchip serial data decoder
- Four channel DMA controller
- UART—full duplex, programmable baud rate (up to 208.3 kBaud)
- 544 kbytes of extended addressing
- Easy interface to National's DASL, 'U' and 'S' transceivers—TP3400, TP3410 and TP3420
- Commercial (0°C to 70°C) Industrial (−40°C to +85°C) and military (−55°C to +125°C) temperature ranges

Block Diagram



TL/DD/8802-1

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Total Allowable Source or Sink Current	100 mA
Storage Temperature Range	−65°C to +150°C
Lead Temperature (Soldering, 10 sec)	300°C
ESD Rating	2000V
V_{CC} with Respect to GND	−0.5V to 7.0V
All Other Pins	(V_{CC} + 0.5)V to (GND − 0.5)V

Note: *Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.*

DC Electrical Characteristics

V_{CC} = 5.0V ±10% unless otherwise specified, T_A = 0°C to +70°C for HPC46400, −40°C to +85°C for HPC36400, −55°C to +125°C for HPC16400

Symbol	Parameter	Test Conditions	Min	Max	Units
I_{CC_1}	Supply Current	V_{CC} = 5.5V, f_{in} = 20.0 MHz (Note 1)		70	mA
		V_{CC} = 5.5V, f_{in} = 2.0 MHz (Note 1)		10	mA
I_{CC_2}	IDLE Mode Current	V_{CC} = 5.5V, f_{in} = 20.0 MHz (Note 1)		10	mA
		V_{CC} = 5.5V, f_{in} = 2.0 MHz (Note 1)		2	mA
I_{CC_3}	HALT Mode Current	V_{CC} = 5.5V, f_{in} = 0 kHz (Note 1)		500	μA
		V_{CC} = 2.5V, f_{in} = 0 kHz (Note 1)		150	μA
INPUT VOLTAGE LEVELS RESET, NMI, CKI AND WO (SCHMITT TRIGGERED)					
V_{IH_1}	Logic High		0.9 V_{CC}		V
V_{IL_1}	Logic Low			0.1 V_{CC}	V
PORT A					
V_{IH_2}	Logic High		2.0		V
V_{IL_2}	Logic Low			0.8	V
ALL OTHER INPUTS					
V_{IH_3}	Logic High		0.7 V_{CC}		V
V_{IL_3}	Logic Low			0.2 V_{CC}	V
I_{LI}	Input Leakage Current			±1	μA
C_I	Input Capacitance	(Note 2)		10	pF
C_{IO}	I/O Capacitance	(Note 2)		20	pF
OUTPUT VOLTAGE LEVELS					
V_{OH_1}	Logic High (CMOS)	I_{OH} = −10 μA (Note 2)	V_{CC} − 0.1		V
V_{OL_1}	Logic Low (CMOS)	I_{OH} = 10 μA (Note 2)		0.1	V
V_{OH_2}	Port A/B Drive, CK2 (A_0–A_{15}, B_{10}, B_{11}, B_{12}, B_{15})	I_{OH} = −7 mA	2.4		V
V_{OL_2}		I_{OL} = 3 mA		0.4	V
V_{OH_3}	Other Port Pin Drive, WO (B_0–B_9, B_{13}, B_{14}, R_0–R_7, D_5, D_7)	I_{OH} = −1.6 mA	2.4		V
V_{OL_3}		I_{OL} = 0.5 mA		0.4	V
V_{OH_4}	ST1 and ST2 Drive	I_{OH} = −6 mA	2.4		V
V_{OL_4}		I_{OL} = 1.6 mA		0.4	V
V_{RAM}	RAM Keep-Alive Voltage	(Note 3)	2.5		V
I_{OZ}	TRI-STATE Leakage Current			±5	μA

Note 1: I_{CC_1}, I_{CC_2}, I_{CC_3} measured with no external drive (I_{OH} and I_{OL} = 0, I_{IH} and I_{IL} = 0). I_{CC_1} is measured with $\overline{RESET}$ = V_{SS}. I_{CC_3} is measured with NMI = V_{CC}. CKI driven to V_{IH_1} and V_{IL_1} with rise and fall times less than 10 ns.

Note 2: These parameters are guaranteed by design and are not tested.

Note 3: Test duration is 100 ms.

AC Electrical Characteristics V_{CC} = 5.0V ±10%, T_A = 0°C to +70°C for HPC46400, −40°C to +85°C for HPC36400, −55°C to +125°C for HPC16400

Symbol	Parameter	Min	Max	Units
f_C = CKI freq.	Operating Frequency	2.0	20	MHz
$t_{C1} = 1/f_C$	Clock Period	50	500	ns
t_{CKIR} (Note 3)	CKI Rise Time		7	ns
t_{CKIF} (Note 3)	CKI Fall Time		7	ns
$[t_{CKIH}/(t_{CKIH} + t_{CKIL})]100$	Duty Cycle	45	55	%
$t_C = 2/f_C$	Timing Cycle	100		ns
$t_{LL} = ½\ t_C - 9$	ALE Pulse Width	41		ns
t_{DC1C2R} (Notes 1, 2)	Delay from CKI Falling Edge to CK2 Rising Edge	0	55	ns
t_{DC1C2F} (Notes 1, 2)	Delay from CKI Falling Edge to CK2 Falling Edge	0	55	ns
$t_{DC1ALER}$ (Notes 1, 2)	Delay from CKI Rising Edge to ALE Rising Edge for CPU Cycles and CKI Falling Edge for DMA Cycles	0	35	ns
$t_{DC1ALEF}$ (Notes 1, 2)	Delay from CKI Rising Edge to ALE Falling Edge for CPU Cycles and CKI Falling Edge for DMA Cycles	0	35	ns
$t_{DC2ALER} = ¼\ t_C + 20$ (Note 2)	Delay from CKI Rising Edge to ALE Rising Edge		45	ns
$t_{DC2ALEF} = ¼\ t_C + 20$ (Note 2)	Delay from CK2 Falling Edge to ALE Falling Edge		45	ns
$t_{ST} = ¼\ t_C - 16$	Address Valid to ALE Falling Edge	9		ns
$t_{WAIT} = t_C$	Wait State Period	100		ns
$f_{XIN} = f_C/19$	External Timer Input Frequency		1052	kHz
$t_{XIN} = t_C$	Pulse Width for Timer Inputs	100		ns
$f_{XOUT} = f_C/16$	Timer Output Frequency		1.25	MHz
$f_{MW} = f_C/19$	External MICROWIRE/PLUS Clock Input Frequency		1.25	MHz
$f_U = f_C/8$	External UART Clock Input Frequency		2.5	MHz

Note 1: Do not design with this parameter unless CKI is driven with an active signal. When using a passive crystal circuit, CKI or CKO **should not** be connected to any external logic since any load (besides the passive components in the crystal circuit) will affect the stability of the crystal unpredictably.

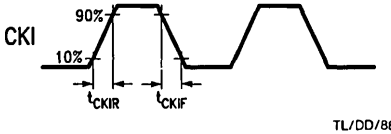
Note 2: These are not directly tested parameters. Therefore the given min/max value cannot be guaranteed. It is, however, derived from measured parameters, and may be used for system design with a very high confidence level.

Note 3: These parameters are guaranteed by design and are not tested.

Note: Measurement of AC specifications is done with external clock drive on CKI with 50% duty cycle. The capacitive load on CKO must be kept below 15 pF else AC measurements will be skewed.

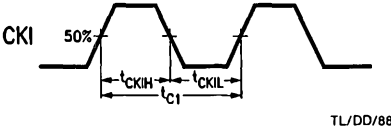
CKI Input Signal Characteristics

Rise/Fall Time



TL/DD/8802-25

Duty Cycle



TL/DD/8802-26

CPU Read Cycle Timing (See *Figure 1*)

Symbol	Parameter	Min	Max	Units
$t_{ARR} = 1/2\ t_C - 20$	ALE Falling Edge to $\overline{RD}$ Falling Edge	30		ns
$t_{RW} = 1/4\ t_C + WS - 10$	$\overline{RD}$ Pulse Width	115		ns
$t_{DR} = t_C - 15$	Data Hold after Rising Edge of $\overline{RD}$	0	85	ns
$t_{ACC} = t_C + WS - 55$ (Note 2)	Address Valid to Input Data Valid		145	ns
$t_{RD} = 1/4\ t_C + WS - 35$	$\overline{RD}$ Falling Edge to Input Data Valid		90	ns
$t_{RDA} = t_C - 5$	$\overline{RD}$ Rising Edge to Address Valid	95		ns
$t_{VP} = 1/4\ t_C - 10$	Address Hold from ALE Falling Edge	15		ns

Note: WS = t_{WAIT} * number of pre-programmed wait states. Minimum and Maximum values are calculated from maximum operating frequency with one (1) wait state pre-programmed.

CPU Write Cycle Timing (See *Figure 2*)

Symbol	Parameter	Min	Max	Units
$t_{ARW} = 1/2\ t_C - 20$	ALE Falling Edge to $\overline{WR}$ Falling Edge	30		ns
$t_{WW} = 3/4\ t_C + WS - 15$	$\overline{WR}$ Pulse Width	160		ns
$t_{HW} = 1/4\ t_C - 5$	Data Hold after Rising Edge of $\overline{WR}$	20		ns
$t_V = 1/2\ t_C + WS - 40$	Data Valid before Rising Edge of $\overline{WR}$	110		ns
$t_{VP} = 1/4\ t_C - 10$	Address Hold from ALE Falling Edge	15		ns

Note: Bus output (Port A) C_L = 100 pF, CK2 output C_L = 50 pF, other outputs C_L = 80 pF. AC Parameters are tested using DC Characteristics and non CMOS outputs.

DMA Read Cycle Timing f_C = 20 MHz (See *Figure 1*)

Symbol	Parameter	Min	Max	Units
$t_{ARR} = 1/2\ t_C - 20$	ALE Falling Edge to $\overline{RD}$ Falling Edge	30		ns
$t_{RW} = 3/2\ t_C - 15$	$\overline{RD}$ Pulse Width	135		ns
$t_{DR} = 3/4\ t_C - 15$	Data Hold After Rising Edge of $\overline{RD}$	0	60	ns
$t_{ACC} = 9/4\ t_C - 75$	Address Valid to Input Data Valid		150	ns
$t_{RD} = 3/2\ t_C - 35$	$\overline{RD}$ Falling Edge to Input Data Valid		115	ns
$t_{RDA} = 3/4\ t_C - 5$	$\overline{RD}$ Rising Edge to Address Valid	95		ns
$t_{VP} = 1/2\ t_C - 10$	Address Hold from ALE Falling Edge	40		ns

Note: Minimum and Maximum values are calculated for maximum operating frequency.

DMA Write Cycle Timing f_C = 20 MHz (See *Figure 2*)

Symbol	Parameter	Min	Max	Units
$t_{ARW} = 1/2\ t_C - 20$	ALE Falling Edge to $\overline{WR}$ Falling Edge	30		ns
$t_{WW} = 3/2\ t_C - 15$	$\overline{WR}$ Pulse Width	135		ns
$t_{HW} = 1/2\ t_C - 15$	Data Hold After Rising Edge of $\overline{WR}$	35		ns
$t_V = 3/2\ t_C - 50$	Data Valid before Rising Edge of $\overline{WR}$	100		ns
$t_{VP} = 1/2\ t_C - 10$	Address Hold from ALE Falling Edge	40		ns

Note: Bus output (Port A) C_L = 100 pF, CK2 output C_L = 50 pF, other outputs C_L = 80 pF. AC Parameters are tested using DC Characteristics and non CMOS outputs.

Ready/Hold Timing

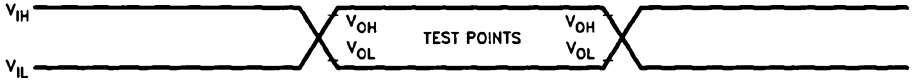
Symbol	Parameter	Min	Max	Units
$t_{DAR} = 1/4\ t_C + WS - 50$	Falling Edge of ALE to Falling Edge of $\overline{RDY}$		75	ns
$t_{RWP} = t_C$	$\overline{RDY}$ Pulse Width	100		ns
$t_{SALE} = 3/4\ t_C + 40$	Falling Edge of $\overline{HLD}$ to Rising Edge of ALE	115		ns
$t_{HWP} = t_C + 10$	$\overline{HLD}$ Pulse Width	110		ns
$t_{HAD} = 3/4\ t_C + 85$	Rising Edge on $\overline{HLD}$ to Rising Edge on $\overline{HLDA}$		160	ns
$t_{HAE} = t_C + 100$	Falling Edge on $\overline{HLD}$ to Falling Edge on $\overline{HLDA}$		200*	ns
$t_{BF} = 1/2\ t_C + 66$	Bus Float after Falling Edge of $\overline{HLDA}$		116 (Due to Emulation)	ns
$t_{BE} = 1/2\ t_C + 66$	Bus Enable before Rising Edge of $\overline{HLDA}$	116		ns

*Note: t_{HAE} may be as long as $(3t_C + 4ws + 72t_C + 90)$ depending on which instruction is being executed, the addressing mode and number of wait states. t_{HAE} maximum value tested is for the optimal case.

MICROWIRE/PLUS Timing

Symbol	Parameter	Min	Max	Units
t_{UWS}	MICROWIRE Setup Time			
Master		100		ns
Slave		20		
t_{UWH}	MICROWIRE Hold Time			
Master		20		ns
Slave		50		
t_{UWV}	MICROWIRE Output Valid Time			
Master			50	ns
Slave			150	

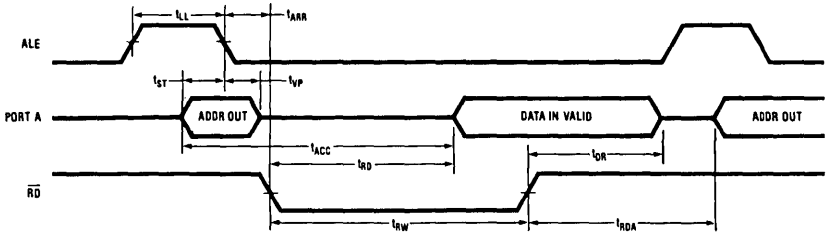
Input and Output for AC Tests



TL/DD/8802-34

Note: AC testing inputs are driven at V_{IH} for a logic "1" and V_{IL} for a logic "0". Output timing measurements are made at V_{OH} for a logic "1" and V_{OL} for a logic "0".

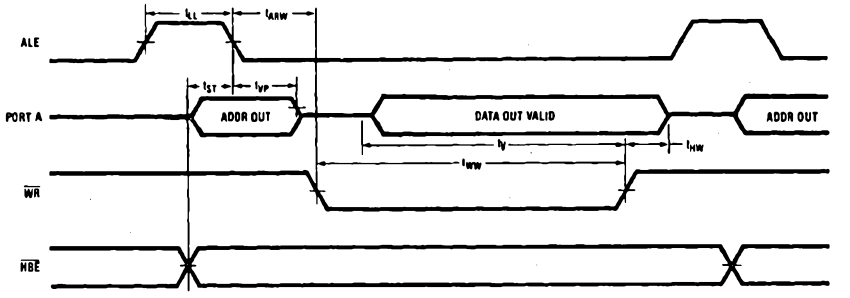
Timing Waveforms



TL/DD/8802-22

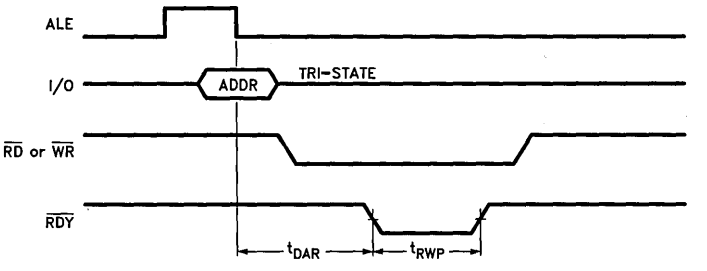
FIGURE 1. CPU and DMA Read Cycles

Timing Waveforms (Continued)



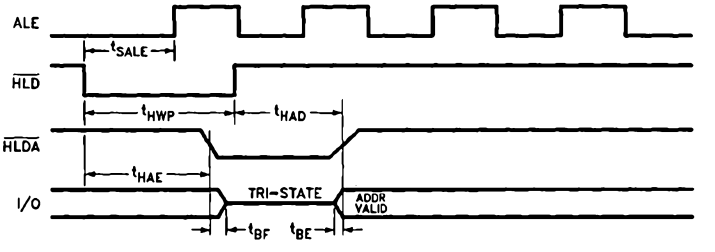
TL/DD/8802-21

FIGURE 2. CPU and DMA Write Cycles



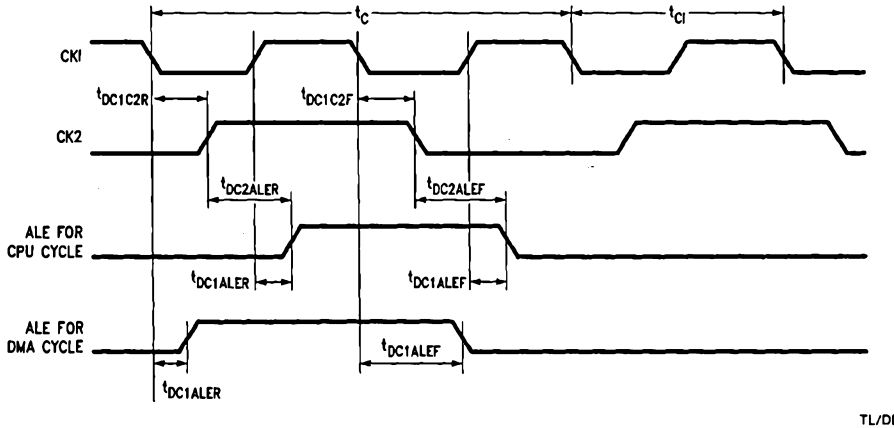
TL/DD/8802-4

FIGURE 3. Ready Mode Timing



TL/DD/8802-5

FIGURE 4. Hold Mode Timing

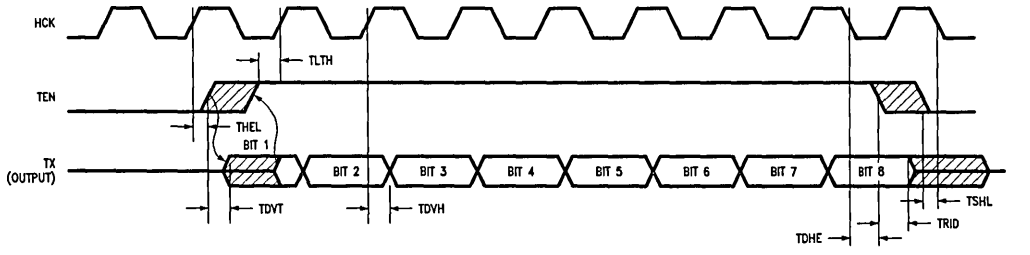


TL/DD/8802-23

FIGURE 5. CKI, CK2 ALE Timing Diagram

Timing Waveforms (Continued)

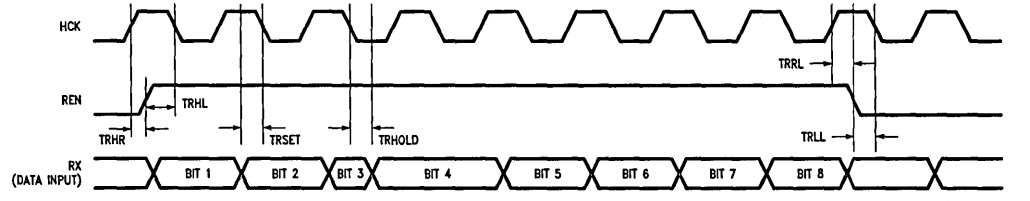
Timing Diagrams for TX using External Enable



TL/DD/8802-28

Symbol	Parameter	Min	Max	Units
THEL	Hold Time of Enable Low to Rising Edge of HCK	5		ns
TDVT	Data Valid Time from Rising Edge of TEN		40	ns
TDVH	Data Valid Time from Rising Edge of HCK		45	ns
TDHE	Hold Time of TEN Falling Edge to HCK Falling Edge	60		ns
TRID	Time From TEN Falling Edge to TRI-STATE® of TX Output		40	ns
TSHL	Setup Time of TEN Falling Edge to HCK Falling Edge	20		ns
TLTH	Setup Time Required for TEN Rising Edge of HCK Rising Edge	80		ns

Timing Diagrams for RX using External Enable



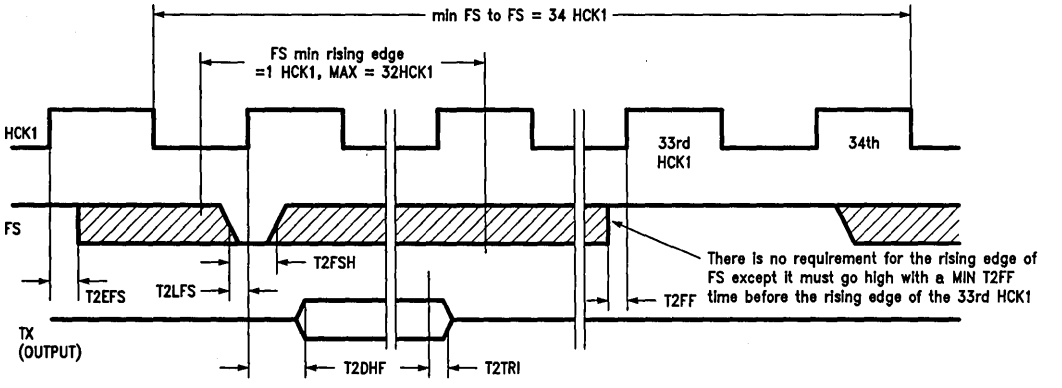
TL/DD/8802-29

Symbol	Parameter	Min	Max	Comments	Units
TRHR	Hold Time of Enable Low to Rising Edge of HCK	5		50% to 50%	ns
TRHL	Setup Time of Rising Edge of REN to Falling Edge of HCK	30		50% to 50%	ns
TRSET	Data Setup Time of RX Data to Falling Edge of HCK	20		50% to 50%	ns
TRHOLD	Data Hold Time of RX Data to Falling Edge of HCK	20		50% to 50%	ns
TRRL	Hold Time of REN High to Rising Edge of HCK	5		50% to 50%	ns
TRLL	Setup Time of REN Low of Falling Edge of HCK	30		50% to 50%	ns

Note: When using RX with normal serial decoder the timings for TRSET and TRHOLD would apply and will have the same values as above.

Timing Waveforms (Continued)

Serial Decoder Timing Diagrams (Mode 2)

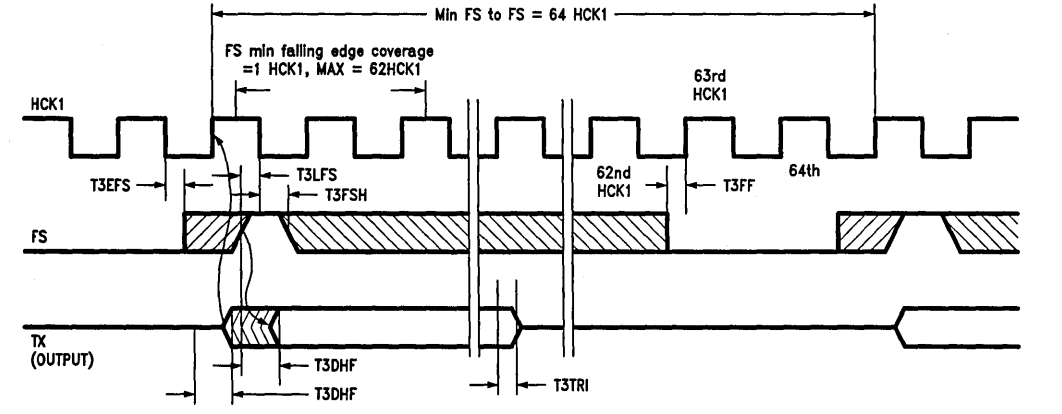


TL/DD/8802-30

Symbol	Parameter	Min	Max	Comments	Units
T2EFS	Delay Time of FS High to Rising Edge of HCK1	10		50% to 50%	ns
T2LFS	Setup Time of FS Low to Rising Edge of HCK1	20		50% to 50%	ns
T2FF	Setup Time of FS High to Rising Edge of the 33rd HCK1	20		50% to 50%	ns
T2FSH	Hold Time of FS Low to Rising Edge of HCK1	20		50% to 50%	ns
T2DHF	Delay From Rising Edge of HCK1 to TX Data Out Valid (First Bit after Valid FS)		50	50% to 0.1 or 0.9 V_{CC}	ns
T2TRI	Delay From Rising Edge of HCK to TRI-STATE of TX Output		40	50% to 0.1 V_{CC} (T0H)	ns
				50% to 0.9 V_{CC} (T0H)	

Note: Receiver operation is guaranteed when min. TRSET and TRHOLD specs are met.

Serial Decoder Timing Diagrams (Mode 3)



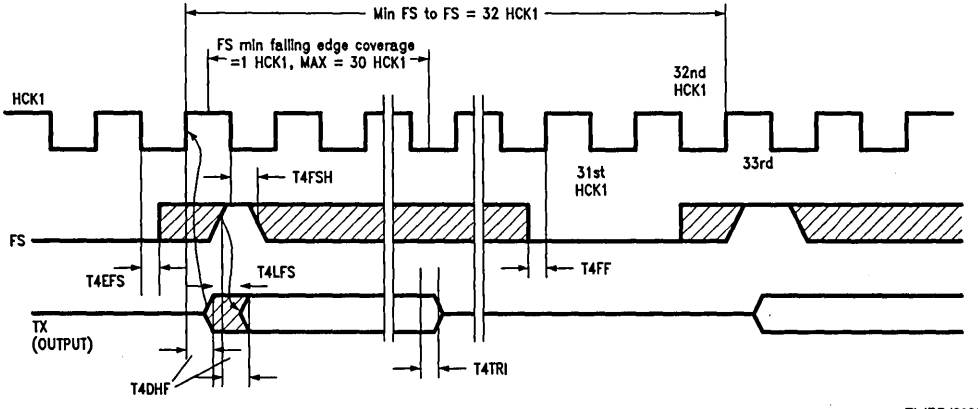
TL/DD/8802-31

Symbol	Parameter	Min	Max	Comments	Units
T3EFS	Delay of Falling Edge of HCK1 to Rising Edge of FS (Early Frame Sync.)	10		50% to 50%	ns
T3LFS	Min Setup Time of FS High to HCK1 Falling Edge to Guarantee RX Operation (Late Frame Sync.)	45		50% to 50%	ns
T3DHF	Delay Time From Rising HCK1 or FS to TX Output Data Valid		50	50% to 0.1 or 0.9 V_{CC}	ns
T3FSH	Hold Time of FS High to Falling Edge of HCK1 (Frame Sync. Hold)	20		50% to 50%	ns
T3FF	Setup Time of FS Low to Rising Edge of 63rd HCK1	20		50% to 50%	ns
T3TRI	Delay From Rising Edge of HCK1 to TRI-STATE of TX Output		40	50% to 0.1 V_{CC} (T0H)	ns
				50% to 0.9 V_{CC} (T1H)	

Note: Receiver operation is guaranteed when TRSET and TRHOLD specs are met.

Timing Waveforms (Continued)

Serial Decoder Timing Diagrams (Mode 4)

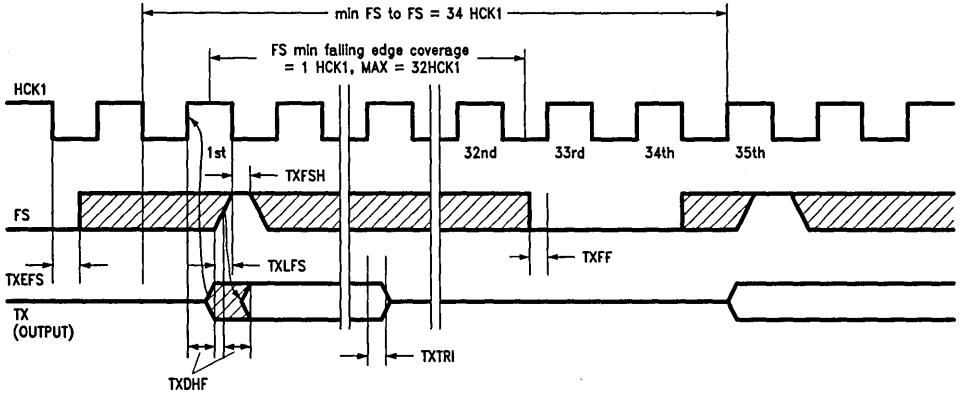


TL/DD/8802-32

Symbol	Parameter	Min	Max	Comments	Units
T4EFS	Delay of Falling Edge of HCK1 to Rising Edge of FS	10		50% to 50%	ns
T4LFS	Min Setup Time FS High to HCK1 Falling Edge to Guarantee RX Operation	45		50% to 50%	ns
T4DHF	Delay Time From Active Edge to TX Output Data Valid		50	50% to 0.1 or 0.9 V_{CC}	ns
T4FSH	Hold Time of FS High to Falling Edge of HCK1	20		50% to 50%	ns
T4FF	Setup Time of FS Low to Rising Edge of 31st HCK1	20		50% to 50%	ns
T4TRI	Delay From Rising Edge of HCK to TRI-STATE of TX Output		40	50% to 0.1 V_{CC} (T0H) 50% to 0.9 V_{CC} (T1H)	ns

Note: Receiver operation is guaranteed when TRSET & TRHOLD specs are met.

Serial Decoder Timing Diagrams (Mode 5,6,7)



X = 5, 6, 7

TL/DD/8802-33

Symbol	Parameter	Min	Max	Comments	Units
TXEFS	Delay of Falling Edge of HCK1 to Rising Edge of FS	10		50% to 50%	ns
TXLFS	Min Setup Time of FS High to HCK1 Falling Edge to Guarantee RX Operation	45		50% to 50%	ns
TXDHF	Delay Time From Active Edge to TX Output Data Valid		50	50% to 0.1 or 0.9 V_{CC}	ns
TXFSH	Hold Time of FS High to Falling Edge of HCK1	20		50% to 50%	ns
TXFF	Setup Time of FS Low to Rising Edge of 33rd HCK1		20	50% to 50%	ns
TXTRI	Delay From Rising Edge of HCK to TRI-STATE of TX Output		40	50% to 0.1 V_{CC} (T0H) 50% to 0.9 V_{CC} (T1H)	ns

Note: Receiver operation is guaranteed when TRSET and TRHOLD specs are met.

Pin Descriptions

I/O PORTS

Port A is a 16-bit multiplexed address/data bus used for accessing external program and data memory. Four associated bus control signals are available on port B. The Address Latch Enable (ALE) signal is used to provide timing to demultiplex the bus. Reading from and writing to external memory are signalled by $\overline{RD}$ and $\overline{WR}$ respectively. External memory can be addressed as either bytes or words with the decoding controlled by two lines, Bus High Byte enable ($\overline{HBE}$) and Address/Data Line 0 (A0).

Port B is a 16-bit port, with 12 bits of bidirectional I/O. Pins B10, B11, B12 and B15 are the control bus signals for the address/data bus. Port B may also be configured via a function register BFUN to individually allow each bidirectional I/O pin to have an alternate function.

B0:	TDX	UART Data Output
B1:	CFLG1	Closing Flag Output for HDLC #1 Transmitter
B2:	CKX	UART Clock (Input or Output)
B3:	T2IO	Timer2 I/O Pin
B4:	T3IO	Timer3 I/O Pin
B5:	SO	MICROWIRE/PLUS Output
B6:	SK	MICROWIRE/PLUS Clock (Input or Output)
B7:	$\overline{HLDA}$	Hold Acknowledge Output
B8:	TS0	Timer Synchronous Output
B9:	TS1	Timer Synchronous Output
B10:	ALE	Address Latch Enable Output for Address/Data Bus
B11:	$\overline{WR}$	Address/Data Bus Write Output
B12:	$\overline{HBE}$	High Byte Enable Output for Address/Data Bus
B13:	TS2	Timer Synchronous Output
B14:	TS3	Timer Synchronous Output
B15:	$\overline{RD}$	Address/Data Bus Read Output

When operating in the extended memory addressing mode, four bits of port B can are used as follows—

B8:	BS0	Memory bank switch output 0 (LSB)
B9:	BS1	Memory bank switch output 1
B13:	BS2	Memory bank switch output 2
B14:	BS3	Memory bank switch output 3 (MSB)

Port I is an 8-bit input port that can be read as general purpose inputs and can also be used for the following functions:

I0:	HCK2	HLDC #2 Clock Input
I1:	NMI	Nonmaskable Interrupt Input
I2:	INT2	Maskable Interrupt/Input Capture
I3:	INT3	Maskable Interrupt/Input Capture
I4:	INT4/RDY	Maskable Interrupt/Input Capture/ Ready Input
I5:	SI	MICROWIRE/PLUS Data Input
I6:	RDX	UART Data Input
I7:	HCK1	HDLC #1 Clock/Serial Decoder Clock Input

Port D is an 8-bit input port that can be read as general purpose inputs and can also be used for the following functions:

D0:	REN1/FS/ RHCK1	Receiver #1 Enable/Serial Decoder Frame Sync Input/Receiver #1 Clock Input
D1:	TEN1	Transmitter #1 Enable Input
D2:	REN2/ RHCK2	Receiver #2 Enable Input/Receiver #2 Clock Input
D3:	TEN2	Transmitter #2 Enable Input
D4:	RX1	Receiver #1 Data Input
D5:	TX1	Transmitter #1 Data Output
D6:	RX2	Receiver #2 Data Input
D7:	TX2	Transmitter #2 Data Output

Note: Any of these pins can be read by software. Therefore, unused functions can be used as general purpose inputs, notably external enable lines when the internal serial decoder is used (see SERIAL DECODER/ENABLE CONFIGURATION REGISTER).

Port R is an 8-bit bidirectional I/O port available for general purpose I/O operations. Port R has a direction register to enable each separate pin to be individually defined as an input or output. It has a data register which contains the value to be output. In addition, the Port R pins can be read directly using the Port R pins address.

Pin Descriptions (Continued)

POWER SUPPLIES

$V_{CC1} + V_{CC2}$ Positive Power Supply (two pins)

GND Ground for On-Chip Logic

DGND Ground for Output Buffers

Note: There are two electrically connected V_{CC} pins on the chip, GND and DGND are electrically isolated. Both V_{CC} pins and both ground pins must be used.

CLOCK PINS

CKI The System Clock Input

CKO The System Clock Output (Inversion of CKI)

Pins CKI and CKO are usually connected across an external crystal.

CK2 Clock Output (CKI divided by 2)

OTHER PINS

$\overline{WO}$ This is an active low open drain output which signals an illegal situation has been detected by the Watch Dog logic.

ST1 Bus Cycle Status Output indicates first opcode fetch.

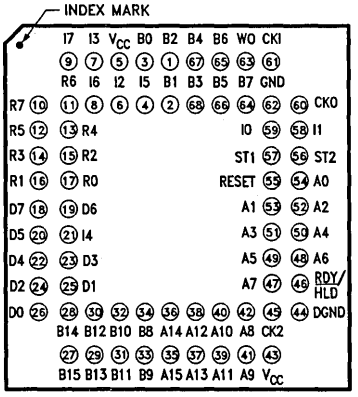
ST2 Bus Cycle Status Output indicates machine states (skip and interrupt).

$\overline{RESET}$ Active low input that forces the chip to restart and sets the ports in a TRI-STATE mode.

RDY/$\overline{HLD}$ Has two uses, selected by a software bit. This pin is either a READY input to extend the bus cycle for slower memories or a HOLD-REQUEST input to put the bus in a high impedance state for external DMA purposes. In the second case the I4 pin becomes the READY input.

Connection Diagram

Pin Grid Array

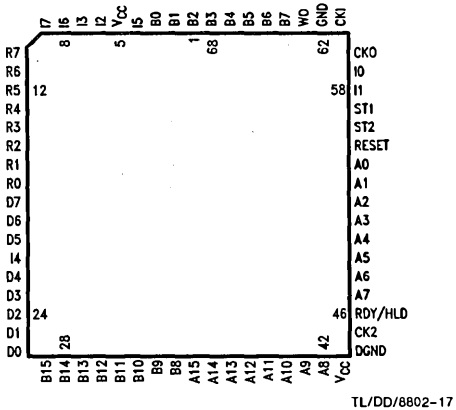


TL/DD/8802-24

Top View

Order Number HPC16400U
See NS Package Number U68A

Plastic, Leadless and Leaded Chip Carriers



TL/DD/8802-17

Top View

Order Number HPC16400E,
HPC16400V or HPC16400L
See NS Package Number E68B, EL68A or V68A

Wait States

The HPC16400 provides four software selectable Wait States that allow access to slower memories. The Wait States are selected by the state of two bits in the PSW register. Additionally, the RDY input may be used to extend the instruction cycle, allowing the user to interface with slow memories and peripherals. The DMA always uses one Wait State, independent of the value selected in the PSW.

Power Save Modes

Two power saving modes are available on the HPC16400: HALT and IDLE. In the HALT mode, all processor activities are stopped. In the IDLE mode, the on-board oscillator and timer T0 are active but all other processor activities are stopped. In either mode, on-board RAM, registers and I/O are unaffected.

HALT MODE

The HPC16400 is placed in the HALT mode under software control by setting bits in the PSW. All processor activities, including the clock and timers, are stopped. In the HALT mode, power requirements for the HPC16400 are minimal and the applied voltage (V_{CC}) may be decreased without altering the state of the machine. There are two ways of exiting the HALT mode: via the $\overline{RESET}$ or the NMI. The $\overline{RESET}$ input reinitializes the processor. Use of the NMI input will generate a vectored interrupt and resume operation from that point with no initialization. The HALT mode can be enabled or disabled by means of a control register HALT enable. To prevent accidental use of the HALT mode the HALT enable register can be modified only once.

IDLE MODE

The HPC16400 is placed in the IDLE mode through the PSW. In this mode, all processor activity, except the on-board oscillator and Timer T0, is stopped. The HPC16400 resumes normal operation upon timer T0 overflow. As with the HALT mode, the processor is also returned to full operation by the $\overline{RESET}$ or NMI inputs, but without waiting for oscillator stabilization.

HPC16400 Interrupts

Complex interrupt handling is easily accomplished by the HPC16400's vectored interrupt scheme. There are eight possible interrupt sources as shown in Table I.

TABLE I. Interrupts

Vector/ Address	Interrupt Source	Arbitration Ranking
FFFF|FFFE	Reset	0
FFFD|FFFC	Nonmaskable Ext (NMI)	1
FFFB|FFFA	External on I2	2
FFF9|FFF8	External on I3	3
FFF7|FFF6	I4 + HDLC/DMA Error	4
FFF5|FFF4	Internal on Timers	5
FFF3|FFF2	Internal on UART	6
FFF1|FFF0	End of Message (EOM)	7

The 16400 contains arbitration logic to determine which interrupt will be serviced first if two or more interrupts occur simultaneously. Interrupts are serviced after the current instruction is completed except for the RESET which is serviced immediately.

The NMI interrupt will immediately stop DMA activity. Byte transfers in progress will finish thereby allowing an orderly transition to the interrupt service vector (see DMA description). The HDLC channels continue to operate, and the user must service data errors that might have occurred during the NMI service routine.

Interrupt Processing

Interrupts are serviced after the current instruction is completed except for the RESET, which is serviced immediately.

RESET is a level-sensitive interrupt. All other interrupts are edge-sensitive. NMI is positive-edge sensitive. The external interrupts on I2, I3 can be software selected to be rising or falling edge.

Interrupt Control Registers

The HPC16400 allows the various interrupt sources and conditions to be programmed. This is done through the various control registers. A brief description of the different control registers is given below.

INTERRUPT ENABLE REGISTER (ENIR)

RESET and the External Interrupt on I1 are non-maskable interrupts. The other interrupts can be individually enabled or disabled. Additionally, a Global Interrupt Enable Bit in the ENIR Register allows the Maskable interrupts to be collectively enabled or disabled. Thus, in order for a particular interrupt to request service, both the individual enable bit and the Global Interrupt bit (GIE) have to be set.

INTERRUPT PENDING REGISTER (IRPD)

The IRPD register contains a bit allocated for each interrupt vector. The occurrence of specified interrupt trigger conditions causes the appropriate bit to be set. There is no indication of the order in which the interrupts have been received. The bits are set independently of the fact that the interrupts may be disabled. IRPD is a Read/Write register. The bits corresponding to the maskable, external interrupts are normally cleared by the HPC16400 after servicing the interrupts.

For the interrupts from the on-board peripherals, the user has the responsibility of resetting the interrupt pending flags through software.

INTERRUPT CONDITION REGISTER (IRCD)

Three bits of the register select the input polarity of the external interrupt on I2, I3, and I4.

Servicing the Interrupts

The Interrupt, once acknowledged, pushes the program counter (PC) onto the stack thus incrementing the stack pointer (SP) twice. The Global Interrupt Enable (GIE) bit is reset, thus disabling further interrupts. The program counter is loaded with the contents of the memory at the vector address and the processor resumes operation at this point. At the end of the interrupt service routine, the user does a RETI instruction to pop the stack, set the GIE bit and return to the main program. The GIE bit can be set in the interrupt service routine to nest interrupts if desired. *Figure 6* shows the Interrupt Enable Logic.

Reset

The RESET input initializes the processor and sets ports A, B (except B12), D and R in the TRI-STATE condition. RESET is an active-low Schmitt trigger input. The processor vectors to FFFF:FFFE and resumes operation at the address contained at that memory location.

Timer Overview

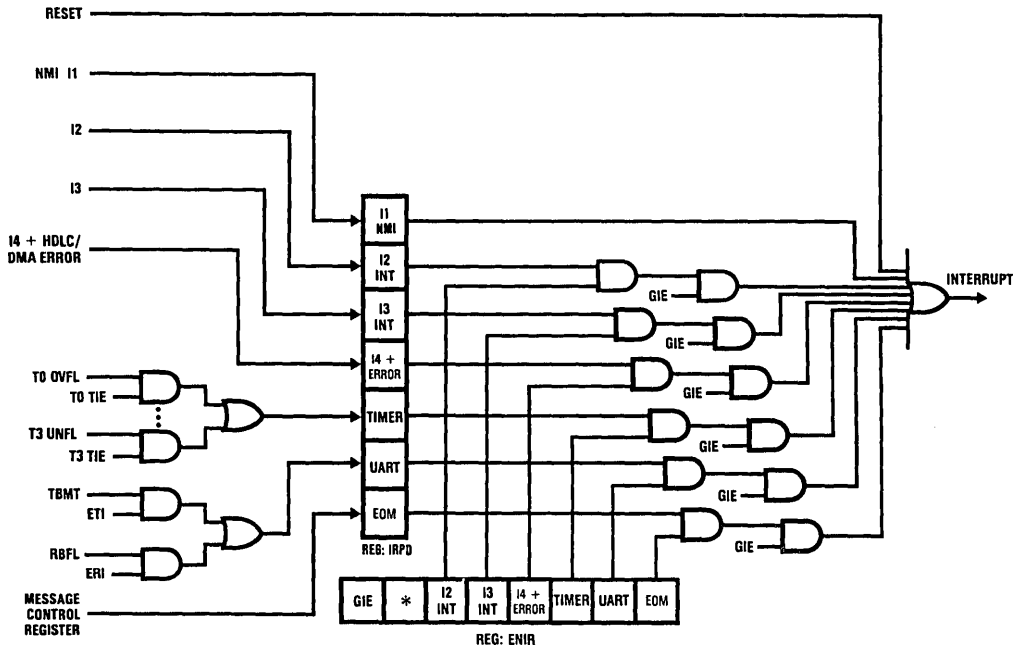
The HPC16400 contains a powerful set of flexible timers enabling the HPC16400 to perform extensive timer functions; not usually associated with microcontrollers.

The HPC16400 contains four 16-bit timers. Three of the timers have an associated 16-bit register. Timer T0 is a free-running timer, counting up at a fixed CKI/16 (Clock Input/16) rate. It is used for Watch Dog logic, high speed event capture, and to exit from the IDLE mode. Consequently, it cannot be stopped or written to under software control. Timer T0 permits precise measurements by means of the capture registers I2CR, I3CR, and I4CR. A control bit in the register T0CON configures timer T1 and its associated register R1 as capture registers I3CR and I2CR. The capture registers I2CR, I3CR, and I4CR respectively, record the value of timer T0 when specific events occur on the interrupt pins I2, I3, and I4. The control register IRCD programs the capture registers to trigger on either a rising edge or a falling edge of its respective input. The specified edge can also be programmed to generate an interrupt (see *Figure 7*).

The timers T2 and T3 have selectable clock rates. The clock input to these two timers may be selected from the following two sources: an external pin, or derived internally by dividing the clock input. Timer T2 has additional capability of being clocked by the timer T3 underflow. This allows the user to cascade timers T3 and T2 into a 32-bit timer/counter. The control register DIVBY programs the clock input to timers T2 and T3 (see *Figure 8*).

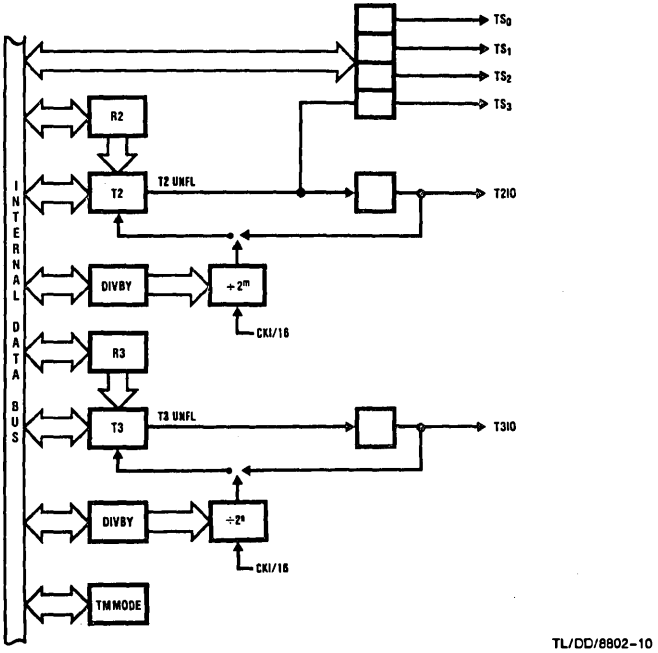
The timers T1 through T3 in conjunction with their registers form Timer-Register pairs. The registers hold the pulse duration values. All the Timer-Register pairs can be read from or written to. Each timer can be started or stopped under software control. Once enabled, the timers count down, and upon underflow, the contents of its associated register are automatically loaded into the timer.

Timer Overview (Continued)



TL/DD/8802-8

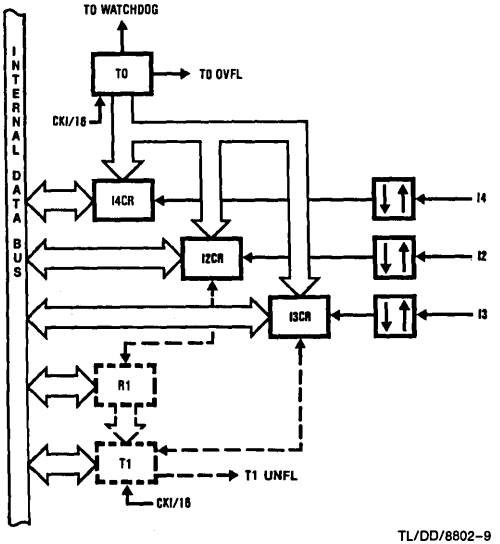
FIGURE 6. Interrupt Enable Logic



TL/DD/8802-10

FIGURE 8. Timers T2–T3 Block

Timer Overview (Continued)



TL/DD/8802-9

FIGURE 7. Timers T0-T1 Block

SYNCHRONOUS OUTPUTS

The flexible timer structure of the HPC16400 simplifies pulse generation and measurement. There are four synchronous timer outputs (TS0 through TS3) that work in conjunction with the timer T2. The synchronous timer outputs can be used either as regular outputs or individually programmed to toggle on timer T2 underflows (see *Figure 8*). Maximum output frequency for any timer output can be obtained by setting timer/register pair to zero. This then will produce an output frequency equal to ½ the frequency of the source used for clocking the timer.

Timer Registers

There are four control registers that program the timers. The divide by (DIVBY) register programs the clock input to timers T2 and T3. The timer mode register (TMMODE) contains control bits to start and stop timers T1 through T3. It also contains bits to latch, acknowledge and enable interrupts from timers T0 through T3.

Timer Applications

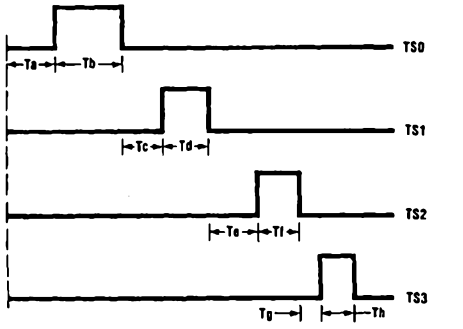
The use of Pulse Width Timers for the generation of various waveforms is easily accomplished by the HPC16400.

Frequencies can be generated by using the timer/register pairs. A square wave is generated when the register value is a constant. The duty cycle can be controlled simply by changing the register value.

TL/DD/8802-12

FIGURE 9. Square Wave Frequency Generation

Synchronous outputs based on Timer T2 can be generated on the 4 outputs TS0-TS3. Each output can be individually programmed to toggle on T2 underflow. Register R2 contains the time delay between events. *Figure 10* is an example of synchronous pulse train generation.



TL/DD/8802-13

FIGURE 10. Synchronous Pulse Generation

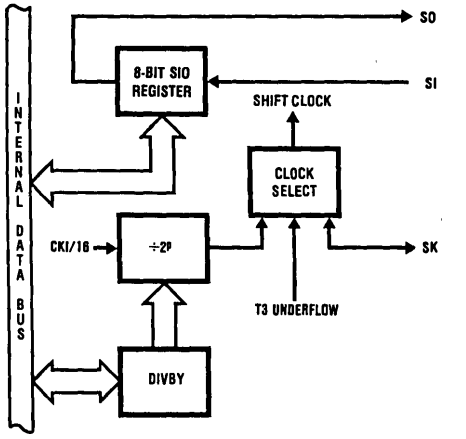
Watch Dog Logic

The Watch Dog Logic monitors the operations taking place and signals upon the occurrence of any illegal activity. The illegal conditions that trigger the Watch Dog logic are potentially infinite loops. Should the Watch Dog register not be written to before Timer T0 overflows twice, or more often than once every 4096 counts, an infinite loop condition is assumed to have occurred. The illegal condition forces the Watch Out (WO) pin low. The WO pin is an open drain output and can be connected to the RESET or NMI inputs or to the users external logic.

MICROWIRE/PLUS

MICROWIRE/PLUS is used for synchronous serial data communications (see *Figure 11*). MICROWIRE/PLUS has an 8-bit parallel-loaded, serial shift register using SI as the input and SO as the output. SK is the clock for the serial shift register (SIO). The SK clock signal can be provided by an internal or external source. The internal clock rate is programmable by the DIVBY register. A DONE flag indicates when the data shift is completed.

The MICROWIRE/PLUS capability enables it to interface with any of National Semiconductor's MICROWIRE peripherals (i.e., ISDN Transceivers, A/D converters, display drivers, EEPROMs).



TL/DD/8802-14

FIGURE 11. MICROWIRE/PLUS

MICROWIRE/PLUS Operation

The HPC16400 can enter the MICROWIRE/PLUS mode as the master or a slave. A control bit in the IRCD register determines whether the HPC16400 is the master or slave. The shift clock is generated when the HPC16400 is configured as a master. An externally generated shift clock on the SK pin is used when the HPC16400 is configured as a slave. When the HPC16400 is a master, the DIVBY register programs the frequency of the SK clock. The DIVBY register allows the SK clock frequency to be programmed in 14 selectable steps from 122 Hz to 1 MHz with CKI at 16 MHz.

The contents of the SIO register may be accessed through any of the memory access instructions. Data waiting to be transmitted in the SIO register is shifted out on the falling edge of the SK clock. Serial data on the SI pin is latched in on the rising edge of the SK clock.

HPC16400 UART

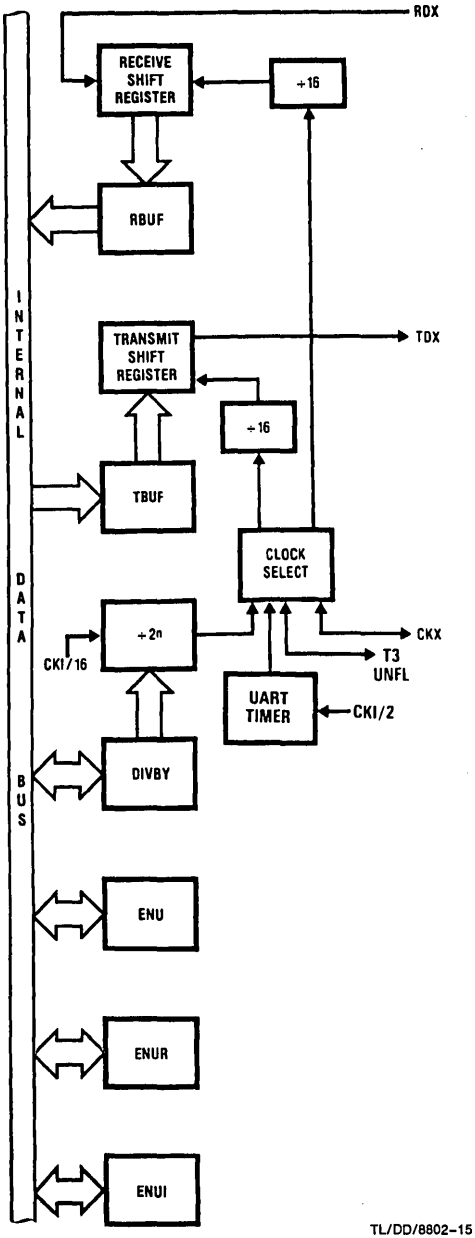
The HPC16400 contains a software programmable UART. The UART (see *Figure 12*) consists of a transmit shift register, a receiver shift register and five addressable registers, as follows: a transmit buffer register (TBUF), a receiver buffer register (RBUF), a UART control and status register (ENU), a UART receive control and status register (ENUR) and a UART interrupt and clock source register (ENUI). The ENU register contains flags for transmit and receive functions; this register also determines the length of the data frame (8 or 9 bits) and the value of the ninth bit in transmission. The ENUR register flags framing and data overrun errors while the UART is receiving. Other functions of the ENUR register include saving the ninth bit received in the data frame and enabling or disabling the UART's Wake-up Mode of operation. The determination of an internal or external clock source is done by the ENUI register, as well as selecting the number of stop bits and enabling or disabling transmit and receive interrupts.

The baud rate clock for the Receiver and Transmitter can be selected for either an internal or external source using two bits in the ENUI register. The internal baud rate is programmed by the DIVBY register, or a special dedicated timer. The baud rate may be selected from a range of 8 baud to 208.3 kbaud. Without having to select a special baud rate crystal, all standard baud rates from 75 baud to 38.4 kbaud can be generated. The external baud clock source comes from the CKX pin. The Transmitter and Receiver can be run at different rates by selecting one to operate from the internal clock and the other from an external source. The special dedicated timer is selected as an external source.

The HPC16400 UART supports two data formats. The first format for data transmission consists of one start bit, eight data bits and one or two stop bits. The second data format for transmission consists of one start bit, nine data bits, and one or two stop bits. Receiving formats differ from transmission only in that the Receiver always requires only one stop bit in a data frame.

UART Wake-up Mode

The HPC16400 UART features a Wake-up Mode of operation. This mode of operation enables the HPC16400 to be networked with other processors. Typically in such environments, the messages consist of addresses and actual data. Addresses are specified by having the ninth bit in the data frame set to 1. Data in the message is specified by having the ninth bit in the data frame reset to 0.



TL/DD/8802–15

FIGURE 12. UART Block Diagram

UART Wake-up Mode (Continued)

The UART monitors the communication stream looking for addresses. When the data word with the ninth bit set is received, the UART signals the HPC16400 with an interrupt. The processor then examines the content of the receiver buffer to decide whether it has been addressed and whether to accept subsequent data.

Programmable Serial Decoder Interface

The programmable serial decoder interface allows the two HDLC channels to be used with devices employing several popular serial protocols for point-to-point and multipoint data exchanges. These protocols combine the 'B' and 'D' channels onto common pins—received data, transmit data, clock and Sync, which normally occurs at an 8 KHz rate and provides framing for the particular protocol.

The decoder uses the serial link clock and Sync signals to generate internal enables for the 'D' and 'B' channels, thereby allowing the HDLC channels to access the appropriate channel data from the multiplexed link.

HDLC Channel Description

HDLC/DMA Structure

HDLC 1		HDLC 2	
HDLC1 Receive	HDLC1 Transmit	HDLC2 Receive	HDLC2 Transmit
DMAR1	DMAT1	DMAR2	DMAT2

GENERAL INFORMATION

Both HDLC channels on the HPC16400 are identical and operate up to 4.65 Mbps. When used in an ISDN basic access application, HDLC channel #1 has been designated for use with the 16 Kbps D-channel or either B channel and HDLC #2 can be used with either of the 64 Kbps B-channels. If the 'D' and 'B' channels are present on a common serial link, the programmable serial decoder interface generates the necessary enable signals needed to access the D and B channel data.

There are two sources for the receive and transmit channel enable signals. They can be internally generated from the serial decoder interface or they can be externally enabled.

LAPD, the Link Access Protocol for the D channel is derived from the X.25 packet switching LAPB protocol. LAPD specifies the procedure for a terminal to use the D channel for the transfer of call control or user-data information. The procedure is used in both point-to-point and point-to-multipoint configurations. On the 16400, the HDLC controller contains user programmable features that allow for the efficient processing of LAPD Information.

HDLC Channel Pin Description

Each HDLC channel has the following pins associated with it.

HCK — HDLC Channel Clock Input Signal.

RX — Receive Serial Data Input. Data latched on the negative HCK edge.

REN/RHCK — HDLC Channel Receiver Enable Input/Receiver Clock Input.

TEN — HDLC Channel Transmitter Enable Input.

TX — Transmit Serial Data Output. Data clocked out on the positive HCK edge. Data (not including CRC) is sent LSB first. TRI-STATE when transmitter not enabled.

HDLC Functional Description

TRANSMITTER DESCRIPTION

Data information is transferred from external memory through the DMA controller into the transmit buffer register from where it is loaded into a 8-bit serial shift register. The CRC is computed and appended to the frame prior to the closing flag being transmitted. Data is output at the TX output pin. If no further transmit commands are given the transmitter sends out continous flags, aborts, or the idle pattern as selected by the control register.

An interrupt is generated when the DMA has transferred the last byte from RAM to the HDLC channel for a particular message or on a transmit error condition. An associated transmit status register will contain the status information indicating the specific interrupt source.

TRANSMITTER FEATURES

Interframe fill: the transmitter can send either continuous '1's or repeated flags or aborts between the closing flag of one packet and the opening flag of the next. When the CPU commands the transmitter to open a new frame, the interframe fill is terminated immediately.

Abort: the 7 '1's abort sequence will be immediately sent on command from the CPU or on an underrun condition in the DMA. If required, it may be followed by a new opening flag to send another packet.

Bit/Byte boundaries: The message length between packet headers may have any number of bits and is not confined to an integral number of bytes. Three bits in the control register are used to indicate the number of valid bits in the last byte. These bits are loaded by the users software.

RECEIVER DESCRIPTION

Data is input to the receiver on the RX pin. The receive clock can be externally input at either the HCK pin or the REN/RHCK pin.

Incoming data is routed through one of several paths depending on whether it is the flag, data, or CRC.

Once the receiver is enabled it waits for the opening flag of the incoming frame, then starts the zero bit deletion, addressing handling and CRC checking. All data between the flags is shifted through two 8-bit serial shift registers before being loaded into the buffer register. The user programmable address register values are compared to the incoming data while it resides in the shift registers. If an address match occurs or if operating in the transparent address recognition mode, the DMA channel is signaled that attention is required and the data is transferred by it to external memory. Appropriate interrupts are generated to the CPU on the reception of a complete frame, or on the occurance of a frame error.

The receive interrupt, in conjunction with status data in the control registers allows interrupts to be generated on the following conditions—frame length error, CRC error, receive error, abort and receive complete.

RECEIVER FEATURES

Flag sharing: the closing flag of one packet may be shared as the opening flag of the next. Receiver will also be able to share a zero between flags—01111101111110 is a valid two flag sequence for receive (not transmit).

HDLC Functional Description (Continued)

Interframe fill: the receiver automatically accepts either repeated flags, repeated aborts, or all '1's as the interframe fill.

Idle: Reception of successive flags as the interframe fill sequence to be signaled to the user by setting the Flag bit in the Receiver Status register.

Short Frame Rejection: Reception of greater than 2 bytes but less than 4 bytes between flags will generate a frame error, terminating reception of the current frame and setting the Frame Error (FER) status bit in the Receive Control and Status register. Reception of less than 2 bytes will be ignored.

Abort: the 7 '1's abort sequence (received with no zero insertion) will be immediately recognized and will cause the receiver to reinitialize and return to searching the incoming data for an opening flag. Reception of the abort will cause the abort status bit in the Interrupt Error Status register to be set and will signal an End of Message (EOMR).

Bit/Byte boundaries: The message length between packet headers may have any number of bits and it is not confined to an integral number of bytes. Three bits in the status register are used to indicate the number of valid bits in the last byte.

Addressing: Two user programmable bytes are available to allow frame address recognition on the two bytes immediately following the opening flag. When the received address matches the programmed value(s), the frame is passed through to the DMA channel. If no match occurs, the received frame address information is disregarded and the receiver returns to searching for the next opening flag and the address recognition process starts anew.

Support is provided to allow recognition of the broadcast address sequence of seven consecutive 1's. Additionally, a transparent mode of operation is available where no address decoding is done.

HDLC INTERRUPT CONDITIONS

The end of message interrupt (EOM) indicates that a complete frame has been received or transmitted by the HDLC controller. Thus, there are four separate sources for this interrupt, two each from each HDLC channel. The Message Control Register contains the pending bits for each source.

The HDLC/DMA error interrupt groups several related error conditions. Error conditions from both transmit/receiver channels can cause this interrupt, and the possible sources each have a status bit in the error status register that is set on the occurrence of an error. The bit must then be serviced by the user.

HDLC CHANNEL CLOCK

Each HDLC channel uses the falling edge of the clock to sample the receive data. Outgoing transmit data is shifted out on the rising edge of the external clock. The maximum data rate when using the externally provided clocks is 4.65 Mb/s.

CYCLIC REDUNDACY CHECK

There are two standard CRC codes used in generating the 16-bit Frame Check Sequence (FCS) that is appended to the end of the data frame. Both codes are supported and the user selects the error checking code to be used through software control (HDLC control reg). The two error checking polynomials available are:

(1) CRC—16 (x16 + x15 + x2 + 1)

(2) CCITT CRC (x16 + x12 + x5 + 1)

SYNCHRONOUS BYPASS MODE

When the BYPAS bit is set in the HDLC control register, all HDLC framing/formatting functions for the specified HDLC channel are disabled.

This allows byte-oriented data to be transmitted and received synchronously thus "bypassing" the HDLC functions.

LOOP BACK OPERATIONAL MODE

The user has the ability, by setting the appropriate bit in the register to internally route the transmitter output to the receiver input, and to internally route the RX pin to the TX pin.

DMA Controller*

GENERAL INFORMATION

The HPC16400 uses Direct Memory Access (DMA) logic to facilitate data transfer between the 2 full Duplex HDLC channels and external packet RAM. There are four DMA channels to support the four individual HDLC channels. Control of the DMA channels is accomplished through registers which are configured by the CPU. These control registers define specific operation of each channel and changes are immediately reflected in DMA operation. In addition to individual control registers, global control bits (MSS and MSSC in Message Control Register) are available so that the HDLC channels may be globally controlled.

The DMA issues a bus request to the CPU when one or more of the individual HDLC channels request service. Upon receiving a bus acknowledge from the CPU, the DMA completes all requests pending and any requests that may have occurred during DMA operation before returning control to the CPU. If no further DMA transfers are pending, the DMA relinquishes the bus and the CPU can again initiate a bus cycle.

Four memory expansion bits have been added for each of the four channels to support data transfers into the expanded memory bank areas.

The DMA has priority logic for a DMA requesting service. The priorities are:

1st priority .....Receiver channel 1

2nd priority .....Transmit channel 1

3rd priority .....Receive channel 2

4th priority .....Transmit channel 2

RECEIVER DMA OPERATION

The receiver DMA consists of a shift register and two buffers. A receiver DMA operation is initiated by the buffer registers. Once a byte has been placed in a buffer register from the HDLC, it generates a request and upon obtaining control of the bus, the DMA places the byte in external memory.

RECEIVER REGISTERS

All the following registers are Read/Write

A. Frame Length Register

This user programmable 16-bit register contains the maximum number of bytes to be placed in a data "block". If this number is exceeded, a Frame Too Long (FTLR1, FTLR2) error is generated. This register is decremented by one each Receiver DMA cycle.

*The specific registers and/or register names may have changed. Please contact the factory for updated information.

DMA Controller (Continued)

B. CNTRL ADDR 1
DATA ADDR 1
CNTRL ADDR 2
DATA ADDR 2

For split frame operation, the CNTRL ADDR register contains the external memory address where the Frame Header (Control & Address fields) are to be stored and the DATA ADDR register contains an equivalent address for the Information field.

For non-split frame operation, the CNTRL and DATA ADDR registers each contain the external memory address for entire frames.

TRANSMITTER DMA OPERATION

The transmitter DMA consists of a shift register and two buffers. A transmitter DMA cycle is initiated by the TX data buffers. The TX data buffers generate a request when either one is empty and the DMA responds by placing a byte in the buffer. The HDLC transmitter can then accept the byte to send when needed, upon which the DMA will issue another request, resulting in a subsequent DMA cycle.

TRANSMITTER REGISTERS

The following registers are Read/Write:

A. Field Address 1 (FA1)
Bytes Field 1 (NBF1)
Field Address (FA2)
Bytes Field 2 (NBF2)

FA1 and FA2 are starting addresses of blocks of information to transmitter.

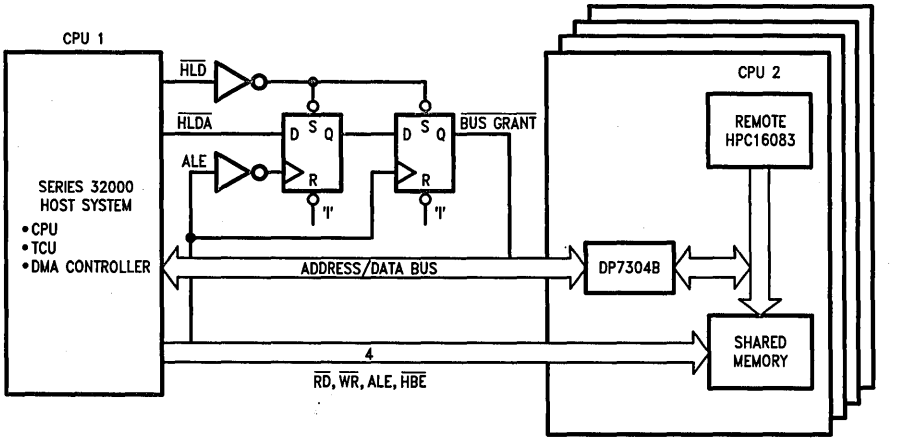
NBF1 and NBF2 are the number of bytes in the block to be transmitted.

Shared Memory Support

Shared memory access provides a rapid technique to exchange data. It is effective when data is moved from a peripheral to memory or when data is moved between blocks of memory. A related area where shared memory access proves effective is in multiprocessing applications where two CPUs share a common memory block. The HPC16400 supports shared memory access with two pins. The pins are the RDY/$\overline{HLD}$ input pin and the $\overline{HLDA}$ output pin. The user can software select either the Hold or Ready function on the RDY/$\overline{HLD}$ pin by the state of a control bit. The $\overline{HLDA}$ output is multiplexed onto port B.

The host uses DMA to interface with the HPC16400. The host initiates a data transfer by activating the $\overline{HLD}$ input of the HPC16400. In response, the HPC16400 places its system bus in a TRI-STATE Mode, freeing it for use by the host. The host waits for the acknowledge signal ($\overline{HLDA}$) from the HPC16400 indicating that the sytem bus is free. On receiving the acknowledge, the host can rapidly transfer data into, or out of, the shared memory by using a conventional DMA controller. Upon completion of the message transfer, the host removes the HOLD request and the HPC16400 resumes normal operations.

Figure 13 illustrates an application of the shared memory interface between the HPC16400 and a Series 32000 system. To insure proper operation, the interface logic shown is recommended as the means for enabling and disabling the user's bus.



TL/DD/8802–16

FIGURE 13. Shared Memory Application: HPC16400 Interface to Series 32000 System

Memory

The HPC16400 has been designed to offer flexibility in memory usage. A total address space of 64 kbytes can be addressed with 256 bytes of RAM available on the chip itself.

Program memory addressing is accomplished by the 16-bit program counter on a byte basis. Memory can be addressed directly by instructions or indirectly through the B, X and SP registers. Memory can be addressed as words or bytes. Words are always addressed on even-byte boundaries. The HPC16400 uses memory-mapped organization to support registers, I/O and on-chip peripheral functions.

The HPC16400 memory address space extends to 64 kbytes and registers and I/O are mapped as shown in Table II.

Extended Memory Addressing

If more than 64k of addressing is desired in a HPC16400 system, on board bank select circuitry is available that allows four I/O lines of Port B (B8, B9, B13, B14) to be used in extending the address range. This gives the user a main routine area of 32k and 16 banks of 32k each for subroutine and data, thus getting a total of 544k of memory.

Note: If all four lines are not needed for memory expansion, the unused lines can be used as general purpose inputs.

The Extended Memory Addressing mode is entered by setting the EMA control bit in the Message Control Register. If this bit is not set, the port B lines (B8, B9, B13, B14) are available as general purpose I/O or synchronous outputs as selected by the BFUN register.

The main memory area contains the interrupt vectors & service routines, stack memory, and common memory for the bank subroutines to use. The 16 banks of memory can contain program or data memory (note: since the on chip resources are mapped into addresses 0000-01FF, the first 512 bytes of each bank are not usable, actual available memory is 536.5k).

TABLE II. Memory Map

FFFF–FFF0 FFEF–FFD0	Interrupt Vectors JSRP Vectors	
FFCF–FFCE : : 0201–0200	External Expansion	USER MEMORY
01FF–01FE : : 01C1–01C0	On Chip RAM	USER RAM
01B8 01B6 01B4 01B2 01B0	Error Status Receiver Status HDLC Cntrl Recr Addr Comp Reg 2 Recr Addr Comp Reg 1	HDLC # 2
01A8 01A6 01A4 01A2 01A0	Error Status Receiver Status HDLC Cntrl Recr Addr Comp Reg 2 Recr Addr Comp Reg 1	HDLC # 1
0195–0194	Watch Dog Register	Watch Dog Logic
0193–0192 0191–0190 018F–018E 018D–018C 018B–018A 0189–0188 0187–0186 0185–0184 0183–0182 0181–0180	T0CON Register TMMODE Register DIVBY Register T3 Timer R3 Register T2 Timer R2 Register I2CR Register/ R1 I3CR Register/ T1 I4CR Register	Timer Block T0–T3
017F–017E 017D–017C	UART Counter UART Register	UART Timer
0179–0178 0177–0176 0175–0174 0173–0172 0171–0170	# Bytes 2 Field Addr 2 # Bytes 1 Field Addr 1 Xmit Cntrl & Status	DMAT # 2 (Xmit)
016B–016A 0169–0168 0167–0166 0165–0164 0163–0162 0161–0160	Frame Length Data Addr 2 Cntrl Addr 2 Data Addr 1 Cntrl Addr 1 Recv Cntrl & Status	DMAR # 2 (Recv)
0159–0158 0157–0156 0155–0154 0153–0152 0151–0150	# Bytes 2 Field Addr 2 # Bytes 1 Field Addr 1 Xmit Cntrl & Status	DMAT # 1 (Xmit)
014B–014A 0149–0148 0147–0146 0145–0144 0143–0142 0141–0140	Frame Length Data Addr 2 Cntrl Addr 2 Data Addr 1 Cntrl Addr 1 Recv Cntrl & Status	DMAR # 1 (Recv)
0128 0126 0124 0122 0120	ENUR Register TBUF Register RBUF Register ENUI Register ENU Register	UART
010E 010C 010A 0106 0104 0102 0100	Port R Pins DIR R Register Port R Data Register Serial Decoder/Enable Configuration Reg Message Pending Message Control Port D Pins	PORTS R & D
00F5–00F4 00F3–00F2 00E3–00E2	BFUN Register DIR B Register Port B	PORT B
00DE 00DD–00DC 00D8 00D6 00D4 00D2 00D0	Microcode ROM Dump Halt Enable Register Port I Input Register SIO Register IRCD Register IRPD Register ENIR Register	PORT CONTROL & INTERRUPT CONTROL REGISTERS
00CF–00CE 00CD–00CC 00CB–00CA 00C9–00C8 00C7–00C6 00C5–00C4 00C3–00C2 00C0	X Register B Register K Register A Register PC Register SP Register (Reserved) PSW Register	HPC16040 CORE REGISTERS
00BF–00BE : : 0001–0000	On Chip RAM	USER RAM

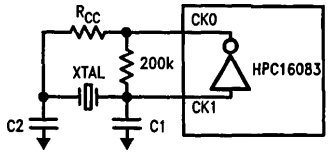
Design Considerations

Designs using the HPC family of 16-bit high speed CMOS microcontrollers need to follow some general guidelines on usage and board layout.

Floating inputs are a frequently overlooked problem. CMOS inputs have extremely high impedance and, if left open, can float to any voltage. You should thus tie unused inputs to V_{CC} or ground, either through a resistor or directly. Unlike the inputs, unused outputs should be left floating to allow the output to switch without drawing any DC current.

To reduce voltage transients, keep the supply line's parasitic inductances as low as possible by reducing trace lengths, using wide traces, ground planes, and by decoupling the supply with bypass capacitors. In order to prevent additional voltage spiking, this local bypass capacitor must exhibit low inductive reactance. You should therefore use high frequency ceramic capacitors and place them very near the IC to minimize wiring inductance.

- Keep V_{CC} bus routing short. When using double sided or multilayer circuit boards, use ground plane techniques.
- Keep ground lines short, and on PC boards make them as wide as possible, even if trace width varies. Use separate ground traces to supply high current devices such as relay and transmission line drivers.
- In systems mixing linear and logic functions and where supply noise is critical to the analog components' performance, provide separate supply buses or even separate supplies.
- When using local regulators, bypass their inputs with a tantalum capacitor of at least 1 μF and bypass their outputs with a 10 μF to 50 μF tantalum or aluminum electrolytic capacitor.
- If the system uses a centralized regulated power supply, use a 10 μF to 20 μF tantalum electrolytic capacitor or a 50 μF to 100 μF aluminum electrolytic capacitor to decouple the V_{CC} bus connected to the circuit board.
- Provide localized decoupling. For random logic, a rule of thumb dictates approximately 10 nF (spaced within 12 cm) per every two to five packages, and 100 nF for every 10 packages. You can group these capacitances, but it's more effective to distribute them among the ICs. If the design has a fair amount of synchronous logic with outputs that tend to switch simultaneously, additional decoupling might be advisable. Octal flip-flop and buffers in bus-oriented circuits might also require more decoupling. Note that wire-wrapped circuits can require more decoupling than ground plane or multilayer PC boards.



TL/DD/8802-35

A recommended crystal oscillator circuit to be used with the HPC is shown below. See table for recommended component values. The recommended values given in the table below have yielded consistent results and are made to match a crystal with a 20 pF load capacitance, with some small allowance for layout capacitance.

A recommended layout for the oscillator network should be as close to the processor as physically possible, entirely within 1" distance. This is to reduce lead inductance from long PC traces, as well as interference from other components, and reduce trace capacitance. The layout should contain a large ground plane either on the top or bottom surface of the board to provide signal shielding, and a convenient location to ground both the HPC, and the case of the crystal.

It is very critical to have an extremely clean power supply for the HPC crystal oscillator. Ideally one would like a V_{CC} and ground plane that provide low inductance power lines to the chip. The power planes in the PC board should be decoupled with three decoupling capacitors as close to the chip as possible. A 1.0 μF, a 0.1 μF, and a 0.001 μF dipped mica or ceramic cap mounted as close to the HPC as is physically possible on the board, using the shortest leads, or surface mount components. This should provide a stable power supply, and noiseless ground plane which will vastly improve the performance of the crystal oscillator network.

HPC Oscillator Table

f_C (MHz)	R_{CC} (Ω)	C1 (pF)	C2 (pF)
2	50	82	100
4	50	62	75
6	50	50	56
8	50	47	50
10	50	39	50
12	0	39	39
14	0	33	39
16	0	33	39
18	0	33	33
20	0	33	33
22	0	27	39
24	0	27	39
26	0	27	33
28	0	27	33
30	0	27	27

Crystal Specifications:
"AT" cut, parallel resonant crystals tuned to the desired frequency with the following specifications are recommended:

Series Resistance < 65Ω

Loading Capacitance: C_L = 20 pF

HPC16400 CPU

The HPC16400 CPU has a 16-bit ALU and six 16-bit registers.

Arithmetic Logic Unit (ALU)

The ALU is 16 bits wide and can do 16-bit add, subtract and shift or logic AND, OR and exclusive OR in one timing cycle. The ALU can also output the carry bit to a 1-bit C register.

Accumulator (A) Register

The 16-bit A register is the source and destination register for most I/O, arithmetic, logic and data memory access operations.

Address (B and X) Registers

The 16-bit B and X registers can be used for indirect addressing. They can automatically count up or down to sequence through data memory.

Boundary (K) Register

The 16-bit K register is used to set limits in repetitive loops of code as register B sequences through data memory.

Stack Pointer (SP) Register

The 16-bit SP register is the stack pointer that addresses the stack. The SP register is incremented by two for each push or call and decremented by two for each pop or return. The stack can be placed anywhere in user memory and be as deep as the available memory permits.

Program (PC) Register

The 16-bit PC register addresses program memory.

Addressing Modes

ADDRESSING MODES—ACCUMULATOR AS DESTINATION

Register Indirect

This is the "normal" mode of addressing for the HPC16400 (instructions are single-byte). The operand is the memory addressed by the B register (or X register for some instructions).

Direct

The instruction contains an 8-bit or 16-bit address field that directly points to the memory for the operand.

Indirect

The instruction contains an 8-bit address field. The contents of the WORD addressed points to the memory for the operand.

Indexed

The instruction contains an 8-bit address field and an 8- or 16-bit displacement field. The contents of the WORD addressed is added to the displacement to get the address of the operand.

Immediate

The instruction contains an 8-bit or 16-bit immediate field that is used as the operand.

Register Indirect (Auto Increment and Decrement)

The operand is the memory addressed by the X register. This mode automatically increments or decrements the X register (by 1 for bytes and by 2 for words).

Register Indirect (Auto Increment and Decrement) with Conditional Skip

The operand is the memory addressed by the B register. This mode automatically increments or decrements the B register (by 1 for bytes and by 2 for words). The B register is then compared with the K register. A skip condition is generated if B goes past K.

ADDRESSING MODES—DIRECT MEMORY AS DESTINATION

Direct Memory to Direct Memory

The instruction contains two 8- or 16-bit address fields. One field directly points to the source operand and the other field directly points to the destination operand.

Immediate to Direct Memory

The instruction contains an 8- or 16-bit address field and an 8- or 16-bit immediate field. The immediate field is the operand and the direct field is the destination.

Double Register Indirect using the B and X Registers

Used only with Reset, Set and IF bit instructions; a specific bit within the 64 kbyte address range is addressed using the B and X registers. The address of a byte of memory is formed by adding the contents of the B register to the most significant 13 bits of the X register. The specific bit to be modified or tested within the byte of memory is selected using the least significant 3 bits of register X.

HPC Instruction Set Description

Mnemonic	Description	Action	
ARITHMETIC INSTRUCTIONS			
ADD	Add	MA + Meml → MA	carry → C
ADDS	Add short imm8	MA + imm8 → MA	carry → C
ADC	Add with carry	MA + Meml + C → MA	carry → C
DADC	Decimal add with carry	MA + Meml + C → MA (Decimal)	carry → C
SUBC	Subtract with carry	MA − Meml + C → MA	carry → C
DSUBC	Decimal subtract w/carry	MA − Meml + C → MA (Decimal)	carry → C
MULT	Multiply (unsigned)	MA*Meml → MA & X, 0 → K, 0 → C	
DIV	Divide (unsigned)	MA/Meml → MA, rem. → X, 0 → K, 0 → C	
DIVD	Divide Double Word (unsigned)	(x8 MA)/Meml → MA, rem → X, 0 → K	carry → C
IFEQ	If equal	Compare MA & Meml, Do next if equal	
IFGT	If greater than	Compare MA & Meml, Do next if MA → Meml	
AND	Logical and	MA and Meml → MA	
OR	Logical or	MA or Meml → MA	
XOR	Logical exclusive-or	MA xor Meml → MA	
MEMORY MODIFY INSTRUCTIONS			
INC	Increment	Mem + 1 → Mem	
DECSZ	Decrement, skip if 0	Mem − 1 → Mem, Skip next if Mem = 0	
BIT INSTRUCTIONS			
SBIT	Set bit	1 → Mem.bit (bit is 0 to 7 immediate)	
RBIT	Reset bit	0 → Mem.bit	
IFBIT	If bit	If Mem.bit is true, do next instr.	
MEMORY TRANSFER INSTRUCTIONS			
LD	Load	Meml → MA	
	Load, incr/decr X	Mem(X) → A, X ± 1 (or 2) → X	
ST	Store to Memory	MA → Mem	
X	Exchange	A ↔ Mem; Mem ↔ Mem	
	Exchange, incr/decr X	A ↔ Mem(X), X ± 1 (or 2) → X	
PUSH	Push Memory to Stack	W → W(SP), SP + 2 → SP	
POP	Pop Stack to Memory	SP − 2 → SP, W(SP) → W	
LDS	Load A, incr/decr B, Skip on condition	Mem(B) → A, B ± 1 (or 2) → B, Skip next if B greater/less than K	
XS	Exchange, incr/decr B, Skip on condition	Mem(B) ↔ A, B ± 1 (or 2) → B, Skip next if B greater/less than K	
REGISTER LOAD IMMEDIATE INSTRUCTIONS			
LD A	Load A immediate	imm → A	
LD B	Load B immediate	imm → B	
LD K	Load K immediate	imm → K	
LD X	Load X immediate	imm → X	
LD BK	Load B and K immediate	imm → B, imm → K	
ACCUMULATOR AND C INSTRUCTIONS			
CLR A	Clear A	0 → A	
INC A	Increment A	A + 1 → A	
DEC A	Decrement A	A − 1 → A	
COMP A	Complement A	1's complement of A → A	
SWAP A	Swap nibbles of A	A15:12 ← A11:8 ← A7:4 ↔ A3:0	
RRC A	Rotate A right thru C	C → A15 → ... → A0 → C	
RLC A	Rotate A left thru C	C ← A15 ← ... ← A0 ← C	
SHR A	Shift A right	0 → A15 → ... → A0 → C	
SHL A	Shift A left	C ← A15 ← ... ← A0 ← 0	
SC	Set C	1 → C	
RC	Reset C	0 → C	
IFC	IF C	Do next if C = 1	
IFNC	IF not C	Do next if C = 0	

HPC Instruction Set Description (Continued)

Mnemonic	Description	Action
TRANSFER OF CONTROL INSTRUCTIONS		
JSRP	Jump subroutine from table	PC → W(SP),SP+2 → SP W(table#) → PC
JSR	Jump subroutine relative	PC → W(SP),SP+2 → SP,PC+# → PC (#is +1024 to −1023)
JSRL	Jump subroutine long	PC → W(SP),SP+2 → SP,PC+# → PC
JP	Jump relative short	PC+# → PC(# is +32 to −31)
JMP	Jump relative	PC+# → PC(#is +256 to −255)
JMPL	Jump relative long	PC+# → PC
JID	Jump indirect at PC + A	PC+A+1 → PC
JIDW		then Mem(PC)+PC → PC
NOP	No Operation	PC ← PC + 1
RET	Return	SP−2 → SP,W(SP) → PC
RETS	Return then skip next	SP−2 → SP,W(SP) → PC, & skip
RETI	Return from interrupt	SP−2 → SP,W(SP) → PC, interrupt re-enabled

Note: W is 16-bit word of memory
MA is Accumulator A or direct memory (8 or 16-bit)
Mem is 8-bit byte or 16-bit word of memory
Meml is 8- or 16-bit memory or 8 or 16-bit immediate data
imm is 8-bit or 16-bit immediate data

Memory Usage

Number Of Bytes For Each Instruction (number in parenthesis is 16-Bit field)

	Using Accumulator A						To Direct Memory			
	Reg Indir. (B)	Reg Indir. (X)	Direct	Indir	Index	Immed.	Direct *	Direct **	Immed. *	Immed. **
LD	1	1	2(4)	3	4(5)	2(3)	3(5)	5(6)	3(4)	5(6)
X	1	1	2(4)	3	4(5)	—	—	—	—	—
ST	1	1	2(4)	3	4(5)	—	—	—	—	—
ADC	1	2	3(4)	3	4(5)	4(5)	4(5)	5(6)	4(5)	5(6)
SBC	1	2	3(4)	3	4(5)	4(5)	4(5)	5(6)	4(5)	5(6)
DADC	1	2	3(4)	3	4(5)	4(5)	4(5)	5(6)	4(5)	5(6)
DSBC	1	2	3(4)	3	4(5)	4(5)	4(5)	5(6)	4(5)	5(6)
ADD	1	2	3(4)	3	4(5)	2(3)	4(5)	5(6)	4(5)	5(6)
MULT	1	2	3(4)	3	4(5)	2(3)	4(5)	5(6)	4(5)	5(6)
DIV	1	2	3(4)	3	4(5)	2(3)	4(5)	5(6)	4(5)	5(6)
IFEQ	1	2	3(4)	3	4(5)	2(3)	4(5)	5(6)	4(5)	5(6)
IFGT	1	2	3(4)	3	4(5)	2(3)	4(5)	5(6)	4(5)	5(6)
AND	1	2	3(4)	3	4(5)	2(3)	4(5)	5(6)	4(5)	5(6)
OR	1	2	3(4)	3	4(5)	2(3)	4(5)	5(6)	4(5)	5(6)
XOR	1	2	3(4)	3	4(5)	2(3)	4(5)	5(6)	4(5)	5(6)

*8-bit direct address
**16-bit direct address

Instructions that modify memory directly

	(B)	(X)	Direct	Indir	Index	B&X
SBIT	1	2	3(4)	3	4(5)	1
RBIT	1	2	3(4)	3	4(5)	1
IFBIT	1	2	3(4)	3	4(5)	1
DECSZ	3	2	2(4)	3	4(5)	
INC	3	2	2(4)	3	4(5)	

Immediate Load Instructions

	Immed.
LD B,*	2(3)
LD X,*	2(3)
LD K,*	2(3)
LD BK,*,*	3(5)

Memory Usage (Continued)

Register Indirect Instructions with Auto Increment and Decrement

Register B With Skip		
	(B+)	(B−)
LDS A,*	1	1
XS A,*	1	1

Register X		
	(X+)	(X−)
LD A,*	1	1
X A,*	1	1

Instructions Using A and C

CLR	A	1
INC	A	1
DEC	A	1
COMP	A	1
SWAP	A	1
RRC	A	1
RLC	A	1
SHR	A	1
SHL	A	1
SC	C	1
RC	C	1
IFC	C	1
IFNC	C	1

Transfer of Control Instructions

JSRP	1
JSR	2
JSRL	3
JP	1
JMP	2
JMPL	3
JID	1
JIDW	1
NOP	1
RET	1
RETS	1
RETI	1

Stack Reference Instructions

	Direct
PUSH	2
POP	2

Code Efficiency

One of the most important criteria of a single chip microcontroller is code efficiency. The more efficient the code, the more features that can be put on a chip. The memory size on a chip is fixed so if code is not efficient, features may have to be sacrificed or the programmer may have to buy a larger, more expensive version of the chip.

The HPC16400 has been designed to be extremely code-efficient. The HPC16400 looks very good in all the standard coding benchmarks; however, it is not realistic to rely only on benchmarks. Many large jobs have been programmed onto the HPC16400, and the code savings over other popular microcontrollers has been considerable.

Reasons for this saving of code include the following:

SINGLE BYTE INSTRUCTIONS

The majority of instructions on the HPC16400 are single-byte. There are two especially code-saving instructions:

JP is a 1-byte jump. True, it can only jump within a range of plus or minus 32, but many loops and decisions are often within a small range of program memory. Most other micros need 2-byte instructions for any short jumps.

JSRP is a 1-byte call subroutine. The user makes a table of his 16 most frequently called subroutines and these calls will only take one byte. Most other micros require two and even three bytes to call a subroutine. The user does not have to decide which subroutine addresses to put into his table; the assembler can give him this information.

EFFICIENT SUBROUTINE CALLS

The 2-byte JSR instructions can call any subroutine within plus or minus 1k of program memory.

MULTIFUNCTION INSTRUCTIONS FOR DATA MOVEMENT AND PROGRAM LOOPING

The HPC16400 has single-byte instructions that perform multiple tasks. For example, the XS instruction will do the following:

1. Exchange A and memory pointed to by the B register
2. Increment or decrement the B register
3. Compare the B register to the K register
4. Generate a conditional skip if B has passed K

The value of this multipurpose instruction becomes evident when looping through sequential areas of memory and exiting when the loop is finished.

BIT MANIPULATION INSTRUCTIONS

Any bit of memory, I/O or registers can be set, reset or tested by the single byte bit instructions. The bits can be addressed directly or indirectly. Since all registers and I/O are mapped into the memory, it is very easy to manipulate specific bits to do efficient control.

DECIMAL ADD AND SUBTRACT

This instruction is needed to interface with the decimal user world.

It can handle both 16-bit words and 8-bit bytes.

The 16-bit capability saves code since many variables can be stored as one piece of data and the programmer does not have to break his data into two bytes. Many applications store most data in 4-digit variables. The HPC16400 supplies 8-bit byte capability for 2-digit variables and literal variables.

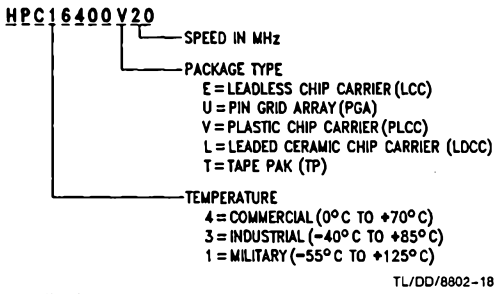
MULTIPLY AND DIVIDE INSTRUCTIONS

The HPC16400 has 16-bit multiply, 16-bit by 16-bit divide, and 32-bit by 16-bit divide instructions. This saves both code and time. Multiply and divide can use immediate data or data from memory. The ability to multiply and divide by immediate data saves code since this function is often needed for scaling, base conversion, computing indexes of arrays, etc.

Part Selection

The HPC family includes devices with many different options and configurations to meet various application needs. The number HPC16400 has been generally used throughout this datasheet to represent the whole family of parts. The following chart explains how to order various options available when ordering HPC family members.

Note: All options may not currently be available.



TL/DD/8802-18

FIGURE 15. HPC Family Part Numbering Scheme

EXAMPLES

HPC46400V20—Commercial temp (0° to +70°C), PLCC

HPC16400L20—Military temp (−55°C to +125°C), LCC

Development Support

DEVELOPMENT SYSTEM

The Microcontroller On Line Emulator (MOLE™) is a low cost development system and emulator for all microcontroller products. These include COPs and the HPC family of products. The development system consists of a BRAIN Board, Personality Board and optional host software.

The purpose of the development system is to provide the user with a tool to write and assemble code, emulate code for the target microcontroller and assist in both software and hardware debugging of the system.

It is a self-contained computer with its own firmware which provides for all system operation, emulation control, communication, PROM programming and diagnostic operations.

It contains three serial ports to optionally connect to a terminal, a host system, a printer or modem, or to connect to other development systems in a multi-development system environment.

The development system can be used in either a stand alone mode or in conjunction with a selected host systems using PC-DOS communicating via a RS-232 port.

HOW TO ORDER

To order a complete development package, select the section for the microcontroller to be developed and order the parts listed.

Development Tools Selection Table

Microcontroller	Order Part Number	Description	Includes	Manual Number
HPC	MOLE-BRAIN	Brain Board	Brain Board Users Manual	420408188-001
	MOLE-HPC-PB2	Personality Board	HPC Personality Board Users Manual	420410477-001
	MOLE-HPC-IBMR	Relocatible Assembler Software for IBM	HPC Software Users Manual and Software Disk PC-DOS Communications Software Users Manual	424410836-001 420040416-001
	MOLE-HPC-IBM-CR	C Compiler for IBM PC	HPC C Compiler Users Manual and Software Disk Assembler Software for IBM MOLE-HPC-IBM	424410883-001
	MOLE-HPC-VMS	Assembler, Loader, Librarian for VAX/VMS	HPC Software User's Manual and 9 Track Tape	424410836-001
	MOLE-HPC-VMS-C	C Compiler for VAX/VMS	HPC Software User's Manual and 9 Track Tape (Includes Assembler)	424410883-001
	424410897-001	Users Manual		

Development Support (Continued)

DIAL-A-HELPER

Dial-A-Helper is a service provided by the Microcontroller Applications Group. Dial-A-Helper is an electronic bulletin board information system and additionally, provides the capability of remotely accessing the MOLE development system at a customer site.

INFORMATION SYSTEM

The Dial-A-Helper system provides access to an automated information storage and retrieval system that may be accessed over standard dial-up telephone lines 24 hours a day. The system capabilities include a MESSAGE SECTION (electronic mail) for communications to and from the Microcontroller Applications Group and a FILE SECTION which consists of several file areas where valuable application software and utilities can be found. The minimum requirement for accessing Dial-A-Helper is a Hayes compatible modem.

If the user has a PC with a communications package then files from the FILE SECTION can be down loaded to disk for later use.

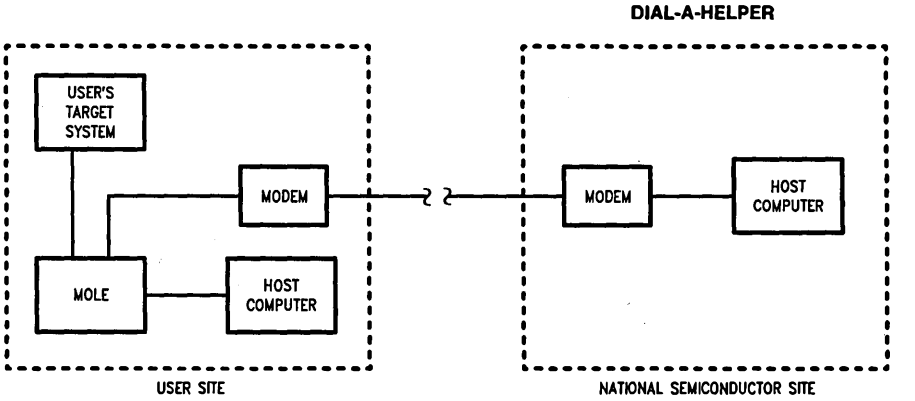
Order P/N: MOLE-DIAL-A-HLP

Information System Package Contains:
- Dial-A-Helper Users Manual
- Public Domain Communications Software

FACTORY APPLICATIONS SUPPORT

Dial-A-Helper also provides immediate factory applications support. If a user is having difficulty in operating a MOLE, he can leave messages on our electronic bulletin board, which we will respond to, or under extraordinary circumstances he can arrange for us to actually take control of his system via modem for debugging purposes.

Voice: (408) 721-5582
Modem: (408) 739-1162
Baud: 300 or 1200 baud
Set-Up: Length: 8-Bit
Parity: None
Stop Bit: 1
Operation: 24 Hrs, 7 Days



TL/DD/8802-20