HPC46400E Slow Clock Mode and HDLC-Based Wakeup

National Semiconductor
Application Note 814
Brian Marley
February 1993

This application note is intended to provide a guide to use of the new Slow Clock power-save feature of the HPC46400E. In particular, this note discusses the motivation, constraints and some recommendations for using it with an HDLC-signalled wakeup.

The Slow Clock feature allows the HPC46400E to enter a low-power mode while maintaining its ability to perform DMA transfers from one of the HDLC receivers at a bit rate of up to 1 megabit/second. The motivation for this feature is to be able to "wake up" the processor to full-speed operation on receipt of a valid HDLC message.

The previously-implemented Power-Save modes (Halt and Idle) do not meet this need, since, in order to guarantee the lowest possible power consumption, they hold certain on-chip peripherals (including the HDLC and DMA sections) in their Reset states. The HDLC channels are therefore unable to trigger an exit from these modes on receipt of a frame; indeed, they are not running at all, and require software set-up on exit from these modes before they can continue operation.

The Slow Clock feature described here is available only as of Revision B of the device. Devices of Revision B or later can be identified by the value "FB" hex, or lower, appearing in the REVREG register. (The convention of decrementing the REVREG value with successive revision steps is due to the fact that the first devices in this family did not have the register at all, and the value read from a non-existent register is "FF" hex in the HPC family.)

THE SLOW CLOCK FEATURE

The Slow Clock feature is controlled only by software, except that a hardware Reset to the HPC will clear it. A program will enter Slow Clock mode by writing a "1" to the SLOW bit, which is bit 7 of the FEXT register (shown below). To come back to full-speed operation, software only has to reset the SLOW bit.

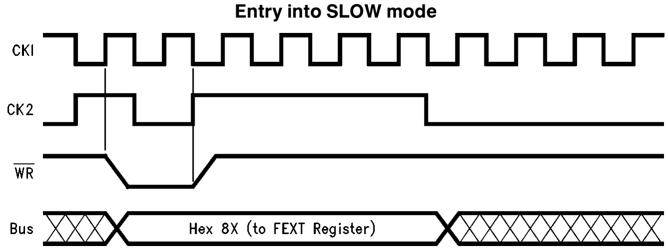
FEXT: Byte at 0110 Hex

7	6	5	4	3	2	1	0
SLOW	(n/a)	(n/a)	(n/a)	(n/a)	(n/a)	SIFFT2	SIFFT1

While the SLOW bit is a "1", the CKI input from the crystal oscillator is divided by four before being used by any on-chip logic (except the CKO pin, of course, which is driving the oscillator). Note that the CK2 clock output is affected by the change; its frequency is divided by four, matching the rate of on-chip events.

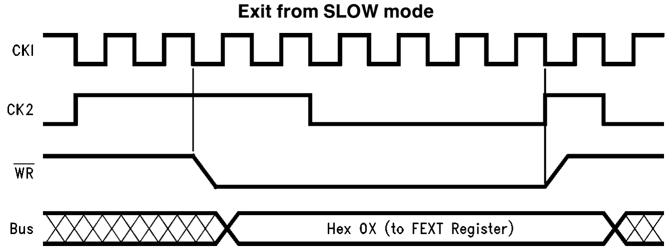
The diagram below shows the timing involved in the entry and exit of Slow Clock mode. The rising edge of the WR/Strobe is the trigger for the switch in frequencies. Note that all Write transfers, both on-chip and off-chip, are visible on the bus pins.

Entry into SLOW mode



TL/DD/11413–1

Exit from SLOW mode



TL/DD/11413–2

 RRD-B5M75/Printed in U. S. A.

Power consumption in Slow Clock mode, with a 20 MHz crystal, drops to slightly under ⅓ of its full-speed value. (This is based on bench measurements as of this writing; data sheet limits have not yet been determined.)

Further division beyond a factor of four was considered, but rejected on the grounds that:

1. The resulting bit rate allowed on the HDLC channels is impacted, and no longer meets the need for a 1 Mbit/second bit rate, and
2. At a division factor of 4, nearly all of the power dissipation is coming from the crystal oscillator itself; additional division would gain little.

While in Slow Clock mode, two important limits change with respect to the HDLC channels:

1. The sampling circuitry which monitors the asynchronous HDLC bit clock inputs (the HCK pins) is running at ¼ of its full-speed rate. Therefore, for reliable sampling, the maximum HDLC clock frequency drops from approximately ¼ of the crystal frequency to $1/16$ of the frequency. For example, at full speed with a 20 MHz crystal attached, the absolute maximum HCK frequency is 5 MHz (specified in the data sheet slightly slower for test integrity). With the SLOW bit set, however, this becomes 1.25 MHz (again minus a small amount, still to be determined at this writing).
2. The bus bandwidth has also been reduced by a factor of four. Any DMA cycles will be impacted by an additional latency due to the fact that any processor bus cycle already in progress will be slower to complete, and the DMA transfers themselves will be slower to complete.

BUS BANDWIDTH IN SLOW MODE

DMA transfers must be considered, since they will be occurring during the first HDLC message received in Slow Mode, before the interrupt is set pending from the Receiver, and therefore before software has had the opportunity to exit from Slow Mode.

The bus bandwidth issue varies somewhat depending on the exact situation. A DMA cycle occupies either four or five bus states (each bus state corresponding to one cycle of CK2), as listed below:

- one Idle bus state (TI), inserted by the DMA section whenever it takes control of the bus from the CPU core,
- one Address state (TA), during which the DMA address is presented,
- one or two Wait states (TW), depending on whether the Ready feature is enabled (this feature allows off-chip logic to request additional Wait states during DMA cycles as well as CPU cycles).
- One Data transfer state (TD), during which data is actually transferred.

At a 1 Mbps HDLC bit rate, bytes become ready for DMA at a rate of one per every 8 μs, corresponding to one DMA cycle per every 20 bus states at 20 MHz in Slow Clock mode. Therefore, DMA activity occupies at most only ¼ of the bus bandwidth in the absence of off-chip Ready or HOLD requests.

DMA LATENCY

As explained above, bus occupancy is not a serious item; however, the latency in granting the bus to the DMA section can be. In the HPC bus, DMA is only granted at a point where an address would have been issued (a TA state). The fact that the bus is idle does not in itself equate to an opportunity for DMA.

There is only a problem if the HPC is allowed to execute Multiply or Divide instructions while in Slow Clock mode. These instructions occupy the bus exclusively for long periods of time (up to 67 bus states between DMA opportunities, assuming one Wait state data memory speed) while performing their calculation within a Read/Modify/Write operation. Even with the on-chip 3-level FIFO in the DMA section, then, use of these instructions will guarantee DMA overruns in SLOW mode (assuming a 20 MHz crystal and 1 Mbps data rate).

The next most significant instructions are the Bit instructions with the addressing mode "X,[B].b", which present a gap in DMA opportunities of seven bus states, regardless of Wait states. Also, some instructions perform a Read/Read/Modify/Write operation, occupying the bus for up to 9 bus states (assuming 1 Wait state data memory). These times are small compared to the amount of time (24 μs: 60 bus states) accounted for by the FIFO buffering in the DMA section; hence, they should be insignificant.

No other instructions occupy the bus in this manner for longer than two bus states between memory accesses. Other impacts could be felt if the Ready feature (Wait states requested by off-chip logic) or the HOLD feature (off-chip DMA) are activated for long periods, but there is no unusual consideration required for these cases.

Note that the extra ALE pulses that appear while performing 16-bit accesses on an eight-bit external bus are not points at which DMA can occur. These are best comprehended as Wait states instead. If you are going to be using the HPC in eight-bit mode, read the section of the User's Manual titled "Bus State Sequences" carefully (this is Section 10.2.1 of the May 1990 edition).

SPECIFIC RECOMMENDATIONS FOR HDLC-BASED WAKEUP

In a relatively "clean" HDLC environment, it is a straightforward process to simply clear the SLOW bit on receipt of an HDLC End of Message (EOM) interrupt. Having processed the HDLC frame, the service routine can then examine its environment and decide whether to re-enter Slow Clock mode before returning.

If, however, the HDLC input is noisy and/or power consumption must absolutely be minimized, it may instead be desirable for the EOM interrupt service routine to check that the message has been received with a good CRC before exiting from Slow Clock mode. The remainder of this discussion will center around this case.

If an HDLC message is received with a bad CRC, certain DMA pointers will have been changed by hardware, and they must be restored to their earlier values before the routine returns. This also requires that the channel be temporarily disabled, to avoid these registers being altered by both software and DMA concurrently (with chaotic results). Disabling the channel creates a short 'dead time' during which a following message will be lost. We will here examine the code required to accomplish this, and the times implied.

The following assumptions are made for purposes of analysis:

- 20 MHz crystal clock.
- Data rate of 1 Mbps.
- The Channel 1 HDLC receiver will be used for the "wake-up" monitoring. This choice is purely arbitrary.
- One Wait state on CPU accesses, two Wait states on DMA accesses (to allow Ready requests from off-chip devices).
- The HPC is placed in a jump-to-self loop while it is in the low-power state. Other instructions besides a jump-to-self are not intended to be forbidden; this assumption is made only for the sake of time analysis, as a simple reasonable case.
- The same interrupt service code will be entered regardless of whether the HPC was in Slow Clock mode when the HDLC frame was received. (This is usually implied if the code is in read-only memory.)

When an End of Message interrupt comes in, the following time constraints apply:

- A second (garbage) frame may trigger another EOM interrupt in as little as 15 μs; DMA might happen within 5 μs. Any processing must take this into account, although any frame received this early cannot be valid. A valid frame could not be received in less than 38 μs.
- If there is a CRC error, the entire interrupt routine will be running in Slow mode. It needs to restore the receiver's DMA pointer registers, clear out errors, and acknowledge the interrupt before returning. It must accomplish this in as little time as possible. Tying this into the consideration above, we want to simplify the handling by guaranteeing that there are no additional EOM interrupts pending.
- Pushing, initializing and restoring processor registers is a detriment to performance in this case. The time penalties involved in performing memory-to-memory instructions are less than the overhead of initializing the pointer registers for more efficient forms. Any saving of the machine state is postponed until after the processor has exited from Slow mode.

We handle this by quickly checking the CRC Error status bit, and turning off the receiver if it is seen and if the processor is in Slow mode. This is accomplished by ANDing the receiver's control register with a mask placed in Basepage RAM before the processor went into Slow mode.

Variables involved are:

PWRPAT	a one-byte Basepage variable that contains 0xEF if the HPC is in Slow Mode, and 0xFF otherwise. It is ANDed with the low byte of the DM1RC register, thus clearing the SSR bit if the processor is in Slow mode.
BUFADDR1, BUFADDR2, BUFADDR3, BUFADDR4	four 16-bit locations in Basepage memory, holding the initial values of the four DMA pointer registers DMR1CA1, DMR1DA1, DMR1CA2 and DMR1DA2. From these values, the DMA registers are restored if a bad HDLC message is received while the HPC is in Slow mode.
FEXT	The new register at address 0x0110, containing the SLOW bit in bit position 7.

Other symbols are register and bit names, as used in the User's Manual.

```
NSC ASMHPC, Version 2.2 (Nov  5 15:25 1991)          EOMWAKE                          20-Feb-92 18:0
EOM Wakeup Code for Interrupt Service                                                        PAGE   1

                                   .title  EOMWAKE,'EOM Wakeup Code for Interrupt Service'

                         ; Registers

      01A8               HD1EST  =        0x01A8:b          ; The HDLC Error register for Channel 1.
      0140               DMR1RCL =        0x0140:b          ; The Receiver Control register for Channel 1
      0110               FEXT    =        0x0110:b          ; The (new) Feature Extension register.
      0104               MSGPND  =        0x0104:b          ; The Messages Pending register.

      0142               DMR1CA1 =        0x0142:w          ; The DMA address registers for Receiver 1.
      0144               DMR1DA1 =        0x0144:w
      0146               DMR1CA2 =        0x0146:w
      0148               DMR1DA2 =        0x0148:w

                         ; Bits

      0007               SLOW    =        7                 ; Slow Mode, in FEXT
      0000               CRCR    =        0                 ; CRC Error flag, in HD1EST

                         ; Variables

      0000                         .sect  DATA,Base

      0000               BUFADDR1: .dsw 1                   ; Four words holding values to restore into
      0002               BUFADDR2: .dsw 1                   ; the DMA pointer registers.
      0004               BUFADDR3: .dsw 1
      0006               BUFADDR4: .dsw 1

      0008               PWRPAT:   .dsb 1                   ; Power Mode indicator pattern: 0xEF = Slow,
                                                            ;  0xFF = Normal.
      0000                         .endsect
```

TL/DD/11413-3

```
                                     .form

                               ;Proposed code in critical path:                              Cycles

0000                                 .sect  CODE,ROM16

0000 60                              JP     .               ;                                    5
                               ;     <interrupt>                                                15
0001                           INTSERV:
0001 B601A810                        IFBIT  CRCR,HD1EST     ; If CRC error, kill receiver       16
0005 81080140D9     R                AND    DMR1RCL,PWRPAT  ;  by turning off SSR bit (iff Slow). 20
000A B601A810                        IFBIT  CRCR,HD1EST     ; Check again and branch to         16
000E 46                              JP     BAD_CRC         ;  decide whether to re-initialize.  5
000F B601101F                        RBIT   SLOW,FEXT       ; If not, then we have a good frame; (15)
0013 942B                            JMP    NORM_FLOW       ;  go to full speed and continue.   ( 7)

0015                           BAD_CRC:
0015 B6011017                        IFBIT  SLOW,FEXT       ; Even if CRC is bad, we want to    16
0019 42                              JP     SLOW_BAD        ;  process normally if not SLOW.     5
001A 9424                            JMP    NORM_FLOW       ;                                   ( 7)

001C                           SLOW_BAD:; DMA is already off; re-initialize pointer and error registers.

                                     ; These assume a worst-case application:  Split-Frame mode
                                     ;  with a non-zero header length.  In any other mode, only
                                     ;  two pointers need to be reloaded (CA1 & DA1 or CA1 & CA2),
                                     ;  since an Interrupt Overrun would kill the channel by
                                     ;  resetting SSR (automatic hardware action).

001C A1000142AB     R                LD     DMR1CA1,BUFADDR1.w;                                 21
0021 A1020144AB     R                LD     DMR1DA1,BUFADDR2.w;                                 21
0026 A1040146AB     R                LD     DMR1CA2,BUFADDR3.w;                                 21
002B A1060148AB     R                LD     DMR1DA2,BUFADDR4.w;                                 21

0030 830001A8D9                      AND    HD1EST,#0       ;(Note AND is faster than LD here.) 18
0035 83000104D9                      AND    MSGPND,#0       ;                                   18

                                     ; Reset and restart receiver, and return from interrupt.

003A 83300140DB                      XOR    DMR1RCL,#0x30   ;                                   18
003F 3E                              RETI                   ;                                    8
                                                            ;                                -----
                                                            ;                              244 CK2
                                                            ;     In Slow Mode (400 ns per CK2):  97.6 us

                                     ; Normal flow, if not CRC error or not SLOW mode.

0040                      NORM_FLOW:
                          ;          .                      ; (Somewhere in here, PWRPAT must be
                          ;          .                      ;  updated to reflect that we are at
                          ;          .                      ;  full speed.)
                          ;          (RETI)

0040                                 .end

**** Errors:    0, Warnings:    0
```

TL/DD/11413-4

CONCLUSIONS

On the second page of the listing, the tabulation to the right of the code is keeping track of the number of CK2 clocks during which the HDLC Receiver is effectively ''off'' while a message with a bad CRC is being processed in Slow Clock mode. A clock count is shown in parentheses when that particular instruction is not executed in this flow. As we can see, the delay is slightly under 100 µs.

The following measurements of overhead can also be made in the receipt of normal messages:

- On normal messages, while not in Slow Clock mode, the first six instructions after the label ''INTSERV'' constitute the additional overhead. This amounts to 79 cycles of CK2, or 7.9 µs. The JP instruction in this flow is actually skipped, but the process of skipping happens to take the same amount of time as a JP instruction.
- On normal messages in Slow Clock mode, the total delay to normal operation can be measured as all instructions executed to the point that the code at label ''NORM__FLOW'' begins execution. Counting also the jump-to-self and the hardware interrupt service time, this amounts to 91 slow clocks (totalling 36.4 µs) plus 8 fast clocks (0.8 µs); 37.2 µs total. Note that the change of clock frequencies takes effect at the beginning of the last clock cycle of the ''RBIT SLOW'' instruction; hence the eight fast clocks instead of just the seven in the ''JMP NORM__FLOW'' instruction.

National Semiconductor Corporation
2900 Semiconductor Drive
P.O. Box 58090
Santa Clara, CA 95052-8090
Tel: 1(800) 272-9959
TWX: (910) 339-9240

National Semiconductor GmbH
Livry-Gargan-Str. 10
D-82256 Fürstenfeldbruck
Germany
Tel: (81-41) 35-0
Telex: 527649
Fax: (81-41) 35-1

National Semiconductor Japan Ltd.
Sumitomo Chemical Engineering Center
Bldg. 7F
1-7-1, Nakase, Mihama-Ku
Chiba-City,
Ciba Prefecture 261
Tel: (043) 299-2300
Fax: (043) 299-2500

National Semiconductor Hong Kong Ltd.
13th Floor, Straight Block,
Ocean Centre, 5 Canton Rd.
Tsimshatsui, Kowloon
Hong Kong
Tel: (852) 2737-1600
Fax: (852) 2736-9960

National Semiconductores Do Brazil Ltda.
Rue Deputado Lacorda Franco
120-3A
Sao Paulo-SP
Brazil 05418-000
Tel: (55-11) 212-5066
Telex: 391-1131931 NSBR BR
Fax: (55-11) 212-1181

National Semiconductor (Australia) Pty, Ltd.
Building 16
Business Park Drive
Monash Business Park
Nottinghill, Melbourne
Victoria 3168 Australia
Tel: (3) 558-9999
Fax: (3) 558-9998