Application Note

AN2310/D
8/2002

MC68HC908QT4 Low Power Application

By: Donnie Garcia
8/16-Bit Products Division
Applications Engineering

Overview

This application note describes a low-power design that uses an MC68HC908QT4 device, a member of the HC08 Family of MCUs from Motorola. Though this member of the QY/QT series contains an A/D module, the points highlighted in this document pertain to all members of the QY/QT series of devices.

Low power consumption is an important consideration in a wide range of applications. Devices such as smoke detectors, wireless keyboards, and remote controls all depend on low power consumption. If power consumption is not considered in the design phase of such applications, the end result will be a product that is unwanted because of its short operational lifetime and high maintenance costs.

The described low-power application uses the auto wakeup feature that allows the MC68HC908QY/QT to bring itself out of stop mode with periodic interrupts. The QY/QT series of MCUs introduces this feature, which is desirable because an external crystal is not needed as a reference for a periodic wakeup. This feature allows the device to reach lower stop mode currents than other parts that require an external crystal to be active during stop mode to provide the same function as the auto wakeup feature.

This document describes the hardware and software used to generate this low-power application. The current profile generated when using this low-power application is quantified and average current estimates are provided for several different cases. This data can be used to calculate estimated battery life for a low-power QY/QT MCU application.

Features

- Includes hardware schematic for the application
- Describes the initialization required to attain low power consumption
- Describes the auto wake up feature
- Contains the software used for the application
- Calculates average current for several different cases

Hardware Requirements

The low-power application has two modes of operation, no alarm mode and alarm mode. In no alarm mode, the application awakes and an A/D reading is taken on PTA5/AD3. The potentiometer connected to PTA5/AD3 is enabled when a 1 is output on PTA3. This configuration allows the potentiometer to be disabled during stop mode (0 output on PTA3). The analog input is used to determine if the application should enter alarm mode. In alarm mode, an LED connected to PTA1 is used to provide an alarm output.

The only other pins that are required are V_{DD} and V_{SS}. A bypass capacitor between these two pins is used to reduce the level of noise on V_{DD}. From the schematic shown in **Figure 1**, it is important to note that most of the pins are left as no connects. In the application software, unused pins are configured as inputs and the internal pullup resistors are enabled. This reduces the need for external hardware on unused pins but it is crucial that these pins are not connected to ground. Connecting any unused pin to ground, while the associated internal pullup resistor is enabled, would cause greater than 100 μA current draw for each pin connected to ground. See **Figure 1**.

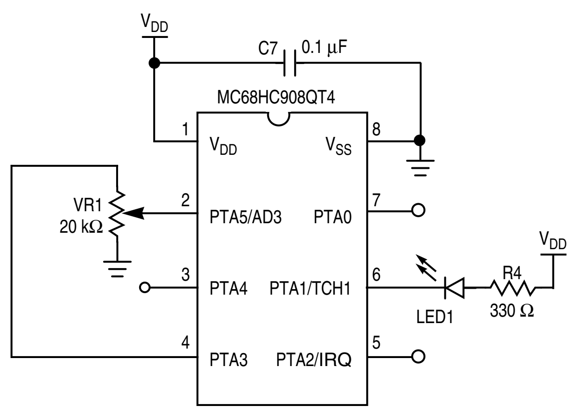

Figure 1. Low-Power Application Schematic

Initialization of Port I/O

As stated in the hardware description, all unused port pins are configured as inputs. Also, the internal pull-up resistors are enabled for the unused pins. With this configuration, there will be no current due to port pins driving signals or floating inputs. It is important to note that PortB is initialized even though this application uses the 8-pin device. This ensures that the lowest possible power consumption can be achieved. See **Figure 2**.

```
;*****  Port A
;*   PTA0,PTA2,PTA4,PTA5 Inputs with Pullups Enabled
;*   PTA1 - LED, active low Initially set to 1
;*   PTA3 - Power to Variable Resister Initially set to 0
;
initPTA:     equ    %00001010     ;initial Port A data
initDDRA:    equ    %00000010     ;initial Port A direction
initPTAPUE:  equ    %00110101     ;initial Port A pullups
;
;*****  Port B
;* All PTB pins are inputs with pullups enabled
;
initPTB:     equ    %00000000     ;initial Port B data
initDDRB:    equ    %00000000     ;initial Port B direction
initPTBPUE:  equ    %11111111     ;initial Port B pullups
```

Figure 2. Port I/O Initialization Code

Initialization of the Configuration Register

The CONFIG1 register must be initialized as shown in **Figure 3**. Stop mode is enabled by the STOP bit in the CONFIG1 register. The SSREC bit is set so that the stop recovery time is 32 BUSCLKX4 cycles. When using the QY/QT clock generator, short stop recovery time can be used because it is not necessary to wait the extra 4096 BUSCLKX4 cycles for clock stabilization. Also note that the LVISTOP bit is set to 0. With this configuration, the LVI module will be disabled in stop mode. In stop mode, there are no clocks and no instructions are being executed so there is no possibility of code runaway that is associated with applications that do not use low-voltage protection. Disabling the LVI in stop mode allows for the lowest possible I_{DDS} to be achieved.

$001F	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	COPRS	LVISTOP	LVIRSTD	LVIPWRD	LVISOR3	SSREC	STOP	COPD
Initialize:	0	0	0	0	1	1	1	1

Figure 3. CONFIG1 Initialization Values

Initialization of Auto Wakeup Interrupt

The COPRS bit in the CONFIG1 register selects the timeout period of the automatic wakeup request (see **Figure 4**). When COPRS is set to 0, the longest period is selected. The longer an application can stay in stop mode, the lower the average current will be. If COPRS was set to 1, then a periodic wakeup would be generated 32 times more often than with the long period selected. This is significant because when the MCU awakes it will draw slightly higher current. Over a long period of time, this would cumulatively affect the overall power consumption. To enable the automatic wakeup interrupts, the AWUIE bit must be set in the keyboard interrupt enable register (KBIER). When this bit is set, the counter that generates the interrupt request will be initiated as the MCU enters stop mode.

$001B	Bit 7	6	5	4	3	2	1	Bit 0
Read:	0	AWUIE	KBIE5	KBIE4	KBIE3	KBIE2	KBIE1	KBIE0
Write:	(unimplemented)							
Initialize:	0	1	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4. Keyboard Interrupt Enable Initialize Values

Software Description

The application software must be initialized for low power as shown above. Also, the A/D module is initialized so a reading can be taken on PTA5/AD3 when the MCU comes out of stop mode. To avoid additional current due to an analog signal on an A/D pin, other modules such as keyboard or timer should be disabled. Disabling these modules will avoid extra supply current associated with the digital buffers for the A/D pin.

For this application, PTA3 is used as an enable signal for the A/D reading. Before a reading, PTA3 must be set so power will be provided to the potentiometer. Before the stop instruction is executed, PTA3 is cleared so that the current consumption of the potentiometer is not active during stop mode.

Out of reset, initialization is completed and the main loop of the software is entered. The main loop consists of an A/D read. If the analog reading is greater than or equal to $80, then the alarm LED is toggled using a software delay (alarm mode is entered). If the analog reading is below $80, then a stop instruction is executed (remain in no alarm mode) and when the next auto wake up interrupt occurs, the main loop is repeated. An A/D reading is taken each time the main loop is executed so when the analog signal returns to below the alarm signal, the application returns to no alarm mode.

Average Current Calculations

Measurements of instantaneous current and the corresponding duration times were taken for different operating conditions on three units at 5 V[1]. For no alarm mode, the measurements showed that in stop mode with auto wake up enabled, the MCU will draw 6 µA. Every 600 ms, the MCU will awake for 25 µs and take an A/D reading. During this time the current increases to 4.85 mA. The total loop time is calculated by adding the duration of the two operating modes. The percent of loop time is calculated by dividing the duration time by the total loop time. The sum of the instantaneous currents multiplied by their corresponding percent of loop time is the average current:

$$0.006\text{ mA}\left(\frac{99.9958}{100}\right) + 4.85\text{mA}\left(\frac{0.0042}{100}\right) = 0.0062\text{ mA}$$

The average current for no alarm mode is calculated to be 6.2 µA. See **Table 1**.

Table 1. Average Current Calculation for No Alarm Mode

Mode	Current (mA)	Duration (ms)	Percent of Loop Time
Stop mode (auto wake up enabled)	0.006	600	99.9958
Wake up and A/D read	4.85	0.025	0.0042
No alarm results Average current Total loop time	0.006201825 —	— 600.025	100

The method described above is used to calculate the average current for alarm mode. In alarm mode, the instantaneous current when the MCU awakes and toggles the alarm LED is 11.25 mA[2]. The duration of this current draw is 350 ms. The calculations show how the longer duration and higher instantaneous current affect the final average current for alarm mode. The average current for alarm mode is calculated to be about 4.15 mA. See **Table 2**.

1. Current measurements would be lower f or 3 V operation.
2. This current could have been reduced by increasing R4 and reducing the brightness of the LED.

Table 2. Average Current Calculation for Alarm Mode

Mode	Current (mA)	Duration (ms)	Percent of Loop Time
Stop mode (auto wake up enabled)	0.006	600	63.1579
Wake up and A/D read and toggle alarm LED	11.25	350	36.8421
Alarm results Average current Total loop time	4.1485 —	— 950	100

Using the calculated average currents for the two modes of operation for our application, we can now calculate the total average current for different test cases. For example, if we expected the application to receive an alarm every 500 minutes, and that this alarm would last 75 seconds, the average current for the system will be 16.5 μA as calculated in **Table 3**.

Table 3. Calculate Average Current for Total System (Case #1)

Mode	Current (mA)	Duration	Percent of Loop Time
Average current (no alarm)	0.0062	500 minutes	99.7506
Average current (with alarm)	4.1485	75 seconds	0.2494
System results (case #1) Average current Total loop time	0.0165 —	— 30075000 ms	100

This calculation is completed using the same method as shown before. It is easy to calculate the estimated average current for a variety of different cases. Below are the calculations for a system that receives many short alarms and for a system that occasionally receives alarms. See **Table 4** and **Table 5**.

Table 4. Calculate Average Current for Total System (Case #2 — Many Short Alarms)

Mode	Current (mA)	Duration	Percent of Loop Time
Average current (no alarm)	0.0062	30 minutes	99.4475
Average current (with alarm)	4.1485	10 seconds	0.5525
System results (case #2) Average current Total loop time	 0.0291 —	 — 1810000 ms	100

Table 5. Calculate Average Current for Total System (Case #3 — Occasionally Receives Alarms)

Mode	Current (mA)	Duration	Percent of Loop Time
Average current (no alarm)	0.0062	3600 minutes	99.9792
Average current (with alarm)	4.1485	45 seconds	0.0208
System results (case #2) Average current Total loop time	 0.0071 —	 — 216045000 ms	100

From these calculations it is evident that the longer the time is between alarms, the lower the average current draw will be. Battery life can be calculated by dividing the battery capacity by the average current draw. In very low-current applications, you should also consider the self discharge characteristics of the selected battery.

Conclusion

The QY/QT is ideal for low-power applications because of the advantage of the auto wakeup feature. The auto wakeup feature does not require an external reference to awake from stop mode. This allows lower stop mode currents that lead to lower average-current consumption. Low power consumption cannot be achieved unless care is taken in initialization. This application note provides a good example of how to attain low power consumption on the MC68HC908QT4.

Low-Power Application Program Listing

```
Metrowerks HC08-Assembler
(c) COPYRIGHT METROWERKS 1987-2002

Loc  Obj. code   Source line
---- ---------   -----------
                 ;.header 'QT4_Low_Power'
                 ;.base 10t
                 ;.pagewidth 130
                 ;.pagelength 90

                 ;*********************************************************************
                 ;*      Copyright (c) Motorola 2002                                  *
                 ;*********************************************************************
                 ;*File name:       QY4_Low_Power.asm Current Release Level:      1.0 *
                 ;*Last Edit Date:  18-July-02        Classification:             ES  *
                 ;*                                                                   *
                 ;*Include Files:   MC68HC908QT4.equ  MC68HC908QT4 MCU definitions    *
                 ;*                                                                   *
                 ;*Assembler:       P&E's CASM08      Version:    3.16                *
                 ;*                                                                   *
                 ;*Target Device:   MC68HC908QT4/QT2/QY4/QY2                          *
                 ;*                                                                   *
                 ;*Documentation:   Low Power App. Note                               *
                 ;*********************************************************************
                 ;* Author:             Donnie Garcia                                 *
                 ;* First Release:      22-July-02                                    *
                 ;*                                                                   *
                 ;* Update History:                                                   *
                 ;*                                                                   *
                 ;* Rev     Date        Author  Description of Change                 *
                 ;* ------  ---------   ------  --------------------------------------*
                 ;*   1.0     7-22-02     DG      Initial Release                     *
                 ;*                                                                   *
                 ;*********************************************************************
                 ;* Motorola reserves the right to make changes without further notice*
                 ;* to any product herein to improve reliability, function, or design.*
                 ;* Motorola does not assume any liability arising out of the         *
                 ;* application or use of any product, circuit, or software described *
                 ;* herein; neither does it convey any license under its patent rights*
                 ;* nor the rights of others.  Motorola products are not designed,    *
                 ;* intended, or authorized for use as components in systems intended *
                 ;* for surgical implant into the body, or other applications intended*
                 ;* to support life, or for any other application in which the failure*
                 ;* of the Motorola product could create a situation where personal   *
                 ;* injury or death may occur.  Should Buyer purchase or use Motorola *
                 ;* products for any such intended or unauthorized application, Buyer *
                 ;* shall indemnify and hold Motorola and its officers, employees,    *
                 ;* subsidiaries, affiliates, and distributors harmless against all   *
                 ;* claims, costs, damages, and expenses, and reasonable attorney     *
                 ;* fees arising out of, directly or indirectly, any claim of personal*
                 ;* injury or death associated with such unintended or unauthorized   *
                 ;* use, even if such claim alleges that Motorola was negligent       *
                 ;* regarding the design or manufacture of the part.                  *
                 ;*                                                                   *
                 ;* Motorola and the Motorola logo are registered trademarks          *
                 ;* of Motorola, Inc.                                                 *
                 ;*********************************************************************
```

```
           ;*  Microcontroller Peripheral Equates  ******************************

           .nolist
                   include "MC68HC908QT4.equ" ; MC68HC908QT4
           .list

0000 000F  initCONFIG1: equ    %00001111     ;Init. for Low Power Consumption
           ;                    ||||||||       CONFIG1 is a write-once register
           ;                    |||||||+-COPD    - 1 disable COP watchdog
           ;                    ||||||+--STOP    - 1 enable STOP instruction
           ;                    |||||+---SSREC   - 1 32 cycle STOP recovery
           ;                    ||||+----LVI5OR3 - 1 set LVI for 5v system
           ;                    |||+-----LVIPWRD - 0 enable power to LVI system
           ;                    ||+------LVIRSTD - 0 enable reset on LVI trip
           ;                    |+-------LVISTOP - 0 disable LVI in STOP mode
           ;                    +--------COPRS   - 0 long COP timeout
0000 00FF  initStack:   equ    RamLast       ;init. stack pointer to last RAM

           ;*****  Port A
           ;*   PTA0,PTA2,PTA4,PTA5 Inputs with Pullups Enabled
           ;*   PTA1 - LED, active low Initially set to 1
           ;*   PTA3 - Power to Variable Resister Initially set to 0
           ;
0000 0002  initPTA:     equ    %00000010     ;initial Port A data
0000 000A  initDDRA:    equ    %00001010     ;initial Port A direction
0000 0035  initPTAPUE:  equ    %00110101     ;initial Port A pullups
           ;
           ;*****  Port B
           ;* All PTB pins are inputs with pullups enabled
           ;
0000 0000  initPTB:     equ    %00000000     ;initial Port B data
0000 0000  initDDRB:    equ    %00000000     ;initial Port B direction
0000 00FF  initPTBPUE:  equ    %11111111     ;initial Port B pullups
           ;
           ;*****  Keyboard Interrupt Module (KBI)
           ;*   Set IMASKK in kbscr to mask interrupts
           ;*   MODEK is 0 for falling edge only
           ;*
0000 0002  initKBSCR:   equ    %00000010     ;initial KBSCR Mask Interrupts
           ;                    ||||||||
           ;                    |||||||+-MODEK   - 0 Interrupts on Falling Edges
           ;                    ||||||+--IMASKK  - 1 Mask Unwanted Interrupts
           ;                    |||||+---ACKK    - 0 Initial value Should be 0
           ;                    ||||+----KEYF    - 0 Initial value Should be 0
           ;                    |||+-----UnImpl  - 0 Unimplemented
           ;                    ||+------UnImpl  - 0 Unimplemented
           ;                    |+-------UnImpl  - 0 Unimplemented
           ;                    +--------UnImpl  - 0 Unimplemented
0000 0040  initKBIER:   equ    %01000000     ;initial KBIER Enable Auto Wakeup
           ;                    ||||||||       CONFIG1 is a write-once register
           ;                    |||||||+-KBIE0   - 0 disable Keyboard Interrupt 0
           ;                    ||||||+--KBIE1   - 0 disable Keyboard Interrupt 1
           ;                    |||||+---KBIE2   - 0 disable Keyboard Interrupt 2
           ;                    ||||+----KBIE3   - 0 disable Keyboard Interrupt 3
           ;                    |||+-----KBIE4   - 0 disable Keyboard Interrupt 4
           ;                    ||+------KBIE5   - 0 disable Keyboard Interrupt 5
           ;                    |+-------AWUIE   - 1 Enable Auto Wakeup Interrupt
           ;                    +--------UnImpl  - 0 Unimplemented

           ;*****  Analog-to-Digital Converter (ADC)
           ;* Used to initialize ADCLK = Bus/2 and select AtoD channel 3 (PTA5)
           ;*
0000 0003  SelectCH3AD: equ    %00000011     ;initial ADSCR Select and Read CH3
0000 0020  Div2ADICLK:  equ    %00100000     ;initial ADICLK = Bus/2
```

```
                ;
                ;*  Global Variables  ************************************************
                ;*
                             org    RamStart      ;begining of RAM
0080            ADDATA       ds     1             ;Variable that contains A/D Reading
                ;
                ;*  Start of Program Space  ******************************************
                ;*
                             org    FlashStart    ;begining of ROM
                ;
                ;*  Power-on Reset  **************************************************
                ;*
                Start:
EE00 4F                  clra                ;Init. Acc. For simulation
EE01 6E 0F 1F                mov    #initCONFIG1,CONFIG1 ;initialize Config1
EE04 45 0100                 ldhx   #initStack+1  ;initialize the stack pointer
EE07 94                      txs
                ;
                ;   Clear all RAM.
                ;
EE08 45 0080                 ldhx   #RamStart     ;point to start of RAM
                ClearRAM:
EE0B 7F                      clr    ,x            ;clear RAM location
EE0C AF 01                   aix    #1            ;advance pointer
EE0E 65 0100                 cphx   #RamLast+1    ;done ?
EE11 26 F8                   bne    ClearRAM      ;loop back if not

                ;
                ;   Initialize Port I/O and Registers
                ;
EE13 6E 02 00                mov    #initPTA,PTA
EE16 6E 35 0B                mov    #initPTAPUE,PTAPUE
EE19 6E 0A 04                mov    #initDDRA,DDRA
EE1C 6E 00 01                mov    #initPTB,PTB
EE1F 6E FF 0C                mov    #initPTBPUE,PTBPUE
EE22 6E 00 05                mov    #initDDRB,DDRB
EE25 6E 02 1A                mov    #initKBSCR,KBSCR
EE28 6E 40 1B                mov    #initKBIER,KBIER
EE2B 6E 20 3F                mov    #Div2ADICLK,ADICLK

EE2E 9A                      cli                  ;enable interrupts
                ;
                ;*  Main Control Loop  ***********************************************
                ;*
                Main:
EE2F 14 1A                   bset   ACKK,KBSCR    ;clear false kbi interrupts
EE31 13 1A                   bclr   IMASKK,KBSCR  ;enable kbi interrupts
EE33 12 00                   bset   PTA1,PTA      ;turn off LED
                AWAKE:
EE35 16 00                   bset   PTA3,PTA  ;Set Power to the POT for reading
EE37 6E 03 3C                mov    #SelectCH3AD,ADSCR ;Select channel3 PTA5
                ADREAD:
EE3A 0F 3C FD                brclr  7,ADSCR,ADREAD ;Wait till read is complete
EE3D B6 3E                   lda    ADR           ;Load A with Analog Data Register
EE3F B7 80                   sta    ADDATA        ;Store current data in ADDATA
                Blink:
EE41 2A 17                   bpl    Sleep         ;Branch to sleep if MSB of data = 0

EE43 13 00                   bclr   PTA1,PTA      ;Turn on the LED
EE45 A6 3C                   lda    #60           ;Delay for on time
EE47 CD EE5F                 jsr    Delay
EE4A 12 00                   bset   PTA1,PTA      ;Turn off the LED
EE4C A6 2C                   lda    #44           ;Delay short for off time
EE4E CD EE5F                 jsr    Delay
```

```
EE51 13 00                      bclr   PTA1,PTA       ;Turn on the LED
EE53 A6 3C                      lda    #60            ;Delay for on time
EE55 CD EE5F                    jsr    Delay
EE58 12 00                      bset   PTA1,PTA       ;Turn off the LED
                 Sleep:
EE5A 17 00                      bclr   PTA3,PTA       ;Clear power to POT for less Idd
EE5C 8E                         stop
EE5D 20 D6                      bra    AWAKE          ;Repeat the Loop

                 ;* End of Main
                 ;*  Delay Subroutine  ************************************************
                 ;* Delay = 4 + Del_var * ((768 * (2+3+3)) + (3 + 3))
                 ;* Delay = 4 + Del_var * 6150
                 ;* This subroutine is a simple software delay loop (x cycles)
                 ;*  Calling convention:
                 ;*      lda     #del_var
                 ;*      jsr     Delay
                 ;*
                 ;*  Returns:    no data
                 ;*  Changes:    ACC, H, X
                 ;
                 Delay:
EE5F 45 0300                    ldhx   #768           ;[3] cycles
EE62 AF FF       Xloop:         aix    #-1            ;[2] cycles
EE64 65 0000                    cphx   #0             ;[3] cycles
EE67 26 F9                      bne    Xloop          ;[3] cycles
EE69 4B F4                      dbnza  Delay          ;[3] cycles
EE6B 81                         rts                   ;[4] cycles
                 ;*********************************************************************

                 ;*  Keyboard Interrupt Service Routine  ******************************
                 ;*
                 ;* This interrupt service routine prompts a new value for the LED
                 ;* toggle frequency
                 ;
                 isrKbd:
EE6C 14 1A                      bset   ACKK,KBSCR     ;acknowledge Auto Wakeup Interrupt
                 Exit:
EE6E 80                         rti
                 ;
                 ;*  Dummy Interrupt  *************************************************
                 ;*
                 Dummy:
EE6F 80                         rti                   ;return
                 ;
                 ;*  Vectors  *********************************************************
                 ;*
                                org    Vadc
FFDE EE6F                       fdb    Dummy          ;ADC vector
FFE0 EE6C                       fdb    isrKbd         ;Keyboard vector
                                org    Vtimov
FFF2 EE6F                       fdb     Dummy          ;Timer overflow vector
                                org    Vtimch1
FFF4 EE6F                       fdb     Dummy          ;Timer channel 1 vector
FFF6 EE6F                       fdb     Dummy          ;Timer channel 0 vector
                                org    Virq
FFFA EE6F                       fdb    Dummy          ;IRQ vector
FFFC EE6F                       fdb     Dummy          ;SWI vector
FFFE EE00                       fdb    Start          ;Reset vector
```

HOW TO REACH US:

USA/EUROPE/LOCATIONS NOT LISTED:

Motorola Literature Distribution;
P.O. Box 5405, Denver, Colorado 80217
1-303-675-2140 or 1-800-441-2447

JAPAN:

Motorola Japan Ltd.; SPS, Technical Information Center,
3-20-1, Minami-Azabu Minato-ku, Tokyo 106-8573 Japan
81-3-3440-3569

ASIA/PACIFIC:

Motorola Semiconductors H.K. Ltd.;
Silicon Harbour Centre, 2 Dai King Street,
Tai Po Industrial Estate, Tai Po, N.T., Hong Kong
852-26668334

TECHNICAL INFORMATION CENTER:

1-800-521-6274

HOME PAGE:

http://www.motorola.com/semiconductors