Chapter 3

Hardware Applications

3.1 I/O Port Usage

The each I/O of Port0, Port1, and Port2 has interrupt capability for the leading edge and trailing edge of an input signal. This has the following advantages:

- ❏ More than one interrupt input is available
- ❏ Eight resp. 24 external events can wake-up from Low Power Modes 3 or 4
- ❏ No glue logic is necessary for most applications: all inputs can be observed without the need of gates connecting them to a single interrupt input.
- ❏ Wake-up is possible out of any input state (high or low)
- ❏ Due to the edge-triggered characteristic of the interrupts, no external switch-off logic is necessary for long-lasting input signals, therefore no multiple interrupt is possible therefore.

3.1.1 General Usage

Six peripheral registers controling the activities of the I/O–Port0 are shown in Table 3–1.

Table 3–1. I/O–Port0 Registers

Register	Usage	State After Reset
Input register	Signals at I/O terminals	Signals at I/O terminals
Output register	Content of output buffer	Unchanged
Direction register	0: Input 1: Output	Reset to input direction
Interrupt flags	0: No interrupt pending 1: Interrupt pending	Set to 0
Interrupt edges	0: Low to high causes Interrupt 1: High to low causes Interrupt	Unchanged
Interrupt enable	0: Disabled 1: Enabled	Set to 0

The interrupt vectors, flags and peripheral addresses of I/O–port 0 are shown in Table 3–2.

Table 3–2. I/O–Port0 Hardware Addresses

Name	Mnemonic	Address	Contents	Vector
Input Register	P0IN	010h	P0IN.7 ... P0IN.0	——
Output Register	P0OUT	011h	P0OUT.7 ... P0OUT.0	
Direction Register	P0DIR	012h	P0DIR.7 ... P0DIR.0	——
Interrupt Flags	P0IFG	013h	P0IFG.7 ... P0IFG.2	0FFE0h
	IFG1.3	002h	P0IFG.1	0FFF8h
	IFG1.2	002h	P0IFG.0	0FFFAh
Interrupt Edges	P0IES	014h	P0IES.7 ... P0IES.0	
Interrupt Enable	P0IE	015h	P0IE.7 ... P0IE.2	
	IE1.3	000h	P0IE.1	
	IE1.2	000h	P0IE.0	

The other I/O–Ports are organized the same way except the following items:

- Port1 and Port2 contain eight equal I/Os, the special hardware for bits 0 and 1 is not implemented. Additionally, the ports have two function select registers, P1SEL and P2SEL .
- Port3 and Port4 do not have interrupt capability and registers P3IFG, P4IFG, P3IES, P4IES, P3IE and P4IE do not exist. Additionally, the ports have two function select registers P3SEL and P4SEL. These registers determine if the normal I/O–Port function is selected (PxSEL.y = 0) or if the terminal is used for a second function (PxSEL.y = 1) (see Table 3–3).

The MSP430C33x uses the two function select registers P3SEL and P4SEL for the following purposes:

- P3SEL.y = 1: The Timer_A I/O functions are selected (see Table 3–3)
- P4SEL.y = 1: The USART functions are selected (see the MSP430x33x Data Sheet, SLAS163)

Table 3–3. Timer_A I/O–Port Selection

P3SEL.y = 0	P3SEL.y = 1 Compare Mode	P3SEL.y = 1 Capture Mode
Port I/O P3.0	Port I/O P3.0	Port I/O P3.0
Port I/O P3.1	Port I/O P3.1	Port I/O P3.1
Port I/O P3.2	Timer Clock input TACLK	Timer Clock input TACLK
Port I/O P3.3	Output TA0	Capture input CCI0A
Port I/O P3.4	Output TA1	Capture input CCI1A
Port I/O P3.5	Output TA2	Capture input CCI2A
Port I/O P3.6	Output TA3	Capture input CCI3A
Port I/O P3.7	Output TA4	Capture input CCI4A

Example 3–1. Using Timer_A in the MSP430C33x System

An MSP430C33x system uses the Timer_A. The Capture/Compare Blocks are used as follows:

- An external clock frequency is used: input at terminal TACLK (P3.2)
- Capture/Compare Block 0: outputs a rectangular signal at terminal TA0 (P3.3)
- Capture/Compare Block 1: outputs a PWM signal at terminal TA1 (P3.4)
- Capture/Compare Block 2: captures the input signal at terminal TA2 (P3.5)

To initialize Port3 for the previous functions the following code line needs to be inserted into the software (for hardware definitions see Section 6.3, *Timer_A*):

```
MOV.B    #TA2+TA1+TA0+TACLK,&P3SEL  ; Initialize Timer I/Os
```

Example 3–2. MSP430C33x System uses the USART Hardware for SCI (UART)

A MSP430C33x system uses the USART hardware for SCI (UART). To initialize terminal P4.7 as URXD and terminal P4.6 as UTXD the following code is used:

```
MOV.B    #URXD+UTXD,&P4SEL          ; Initialize SCI I/Os
```

Example 3–3. The I/O–ports P0.0 to P0.3 are used for input only.

The I/O–ports P0.0 to P0.3 are used for input only. Terminals P0.4 to P0.7 are outputs and initially set low. The conditions are:

```
P0.0     Every change is counted
P0.1     Any high–to–low change is counted
P0.2     Any low–to–high change is counted
P0.3     Every change is counted
;
; RAM definitions
;
         .BSS     P0_0CNT,2                ; Counter for P0.0
         .BSS     P0_1CNT,2                ; Counter for P0.1
         .BSS     P0_2CNT,2                ; Counter for P0.2
         .BSS     P0_3CNT,2                ; Counter for P0.3
;
; Initialization for Port0
;
         MOV.B    #000h,&P0OUT             ; Output register low
         MOV.B    #0F0h,&P0DIR             ; P0.4 to P0.7 outputs
         MOV.B    #00Bh,&P0IES             ; P0.0 to P0.3 Hi–Lo, P0.2 Lo–Hi
         MOV.B    #00Ch,&P0IE              ; P0.2 to P0.3 interrupt enable
         BIS.B    #00Ch,&IE1               ; P0.0 to P0.1 interrupt enable
;
; Interrupt handler for P0.0. Every change is counted
;
P0_0HAN  INC      P0_0CNT                  ; Flag is reset automatically
         XOR.B    #1,&P0IES                ; Change edge select
         RETI
;
; Interrupt handler for P0.1. Any Hi–Lo change is counted
;
P0_1HAN  INC      P0_1CNT                  ; Flag is reset automatically
         RETI
;
; Interrupt handler for P0.2 and P0.3
```

```
; The flags of all read transitions are reset. Transitions
; occurring during the interrupt routine cause interrupt after
; the RETI
;
P0_23HAN PUSH     R5                                ; Save R5
         MOV.B    &P0FLG,R5                         ; Copy interrupt flags
         BIC.B    R5,&P0FLG                         ; Reset read flags
         BIT      #4,R5                             ; P0.2 flag to carry
         ADC      P0_2CNT                           ; Add carry to counter
         BIT      #8,R5                             ; P0.3 flag to carry
         JNC      L$304
         INC      P0_3CNT                           ; P0.3 changed
         XOR.B    #8,P0IES                          ; Change edge select
L$304    POP      R5                                ; Restore R5
         RETI
;
         .SECT    "INT_VECT",0FFF8h
         .WORD    P0_1HAN                           ; P0.1 INTERRUPT VECTOR;
         .WORD    P0_0HAN                           ; P0.0 INTERRUPT VECTOR;
;
         .SECT    "INT_VECT1",0FFE0h
         .WORD    P0_23HAN                          ; P0.2/7 INTERRUPT VECTOR
```

3.1.2 Zero Crossing Detection

With the external components shown in Figure 3–1 it is possible to build a zero crossing input for the MSP430. The components shown are designed for an external voltage ac = 230 V. With a circuit capacitance (wiring, diodes) of C1 = 30 pF as shown in Figure 3–1, the following delays occur (all values for ac = 230 V, f = 50 Hz, V_{CC} = +5 V, timing is in µs):

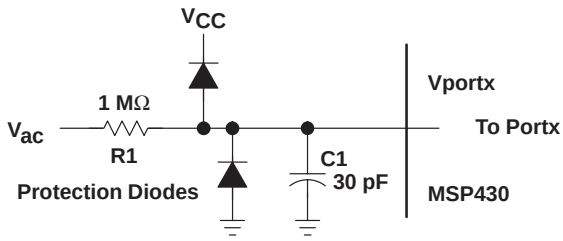


Figure 3–1. MSP430 Input for Zero-Crossing

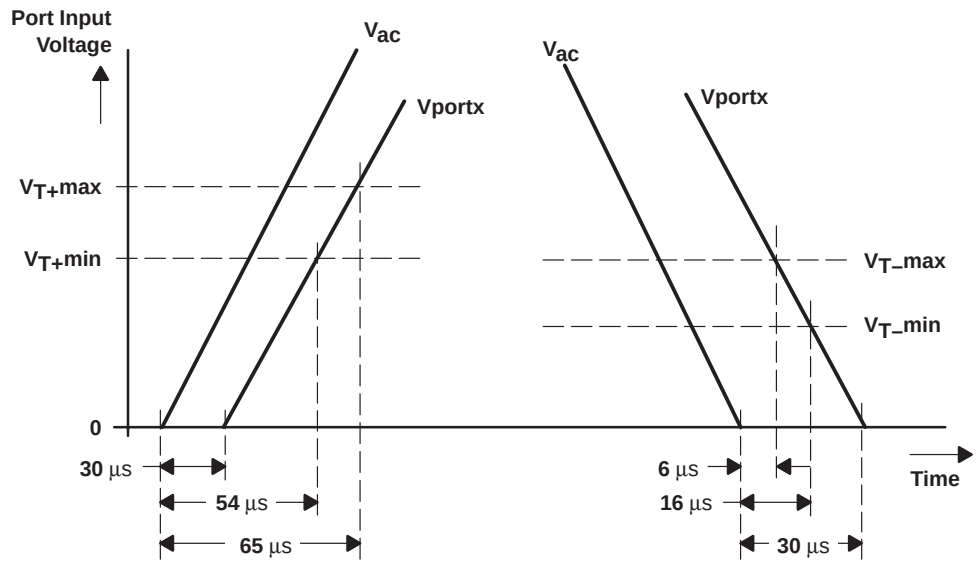


Figure 3–2. Timing for the Zero Crossing

Delay caused by RC (1 MΩ x 30 pF): 30 µs or 0.54° (same value for leading and trailing edges).

Delay caused by input thresholds:

Leading edge: 24 µs to 35 µs. (V_{T+} = 2.3 V to 3.4 V)
Trailing edge: 14 µs to 24 µs. (V_{T-} = 1.4 V to 2.3 V)

The resulting delays are:

Leading edge: 54 µs to 65 µs.
Trailing edge: 6 µs to 16 µs.

These small deviations do not play a role for 50 Hz or 60 Hz phase control applications with TRIACs. If other input conditions than 230 V and 50 Hz are used then the resulting delays can be calculated with the following formulas:

$$t_D = \frac{V_T}{S_V};\ S_V = \frac{d(U \times \sin\omega t)}{dt} = U \times \omega \times \cos\omega t$$

Where:

t_D	Delay time caused by the input threshold voltage	[s]
V_T	Input threshold voltage	[V]
S_V	Slope of the input voltage	[V/s]
ω	Angular frequency $2\pi f$	[1/s]
U	Peak value of the input voltage U_{ac}	[V]

For t = 0 (zero crossing time) the previous equation becomes:

$$t_D = \frac{V_T}{U \times \omega \times 1} = \frac{V_T}{U \times \omega}$$

3.1.3 Output Buffering

The outputs of the MSP430 (P0.x, P1.x, P2.x, P3.x, P4.x, Ox) have nominal internal resistances depending on the supply voltage, V_{CC}:

V_{CC} = 3 V: Max. 333 Ω (ΔV = 0.4V max. @ 1.2mA)

V_{CC} = 5 V: Max. 266 Ω (ΔV = 0.4V max. @ 1.5mA)

These internal resistances are non-linear and are valid only for small output currents (see the previous text). If larger currents are drawn, saturation effects will limit the output current.

These outputs are intended for driving digital inputs and gates and normally have too high an impedance level for other applications, such as the driving of relays, lines, etc. If output currents greater than the previously mentioned ones are needed then output buffering is necessary. Figure 3–3 shows some of the possibilities. The resistors shown in Figure 3–3 for the limitation of the MSP430 output current are minimum values. The application is designed for V_{CC} = 5 V. The values shown in brackets are for V_{CC} = 3 V.

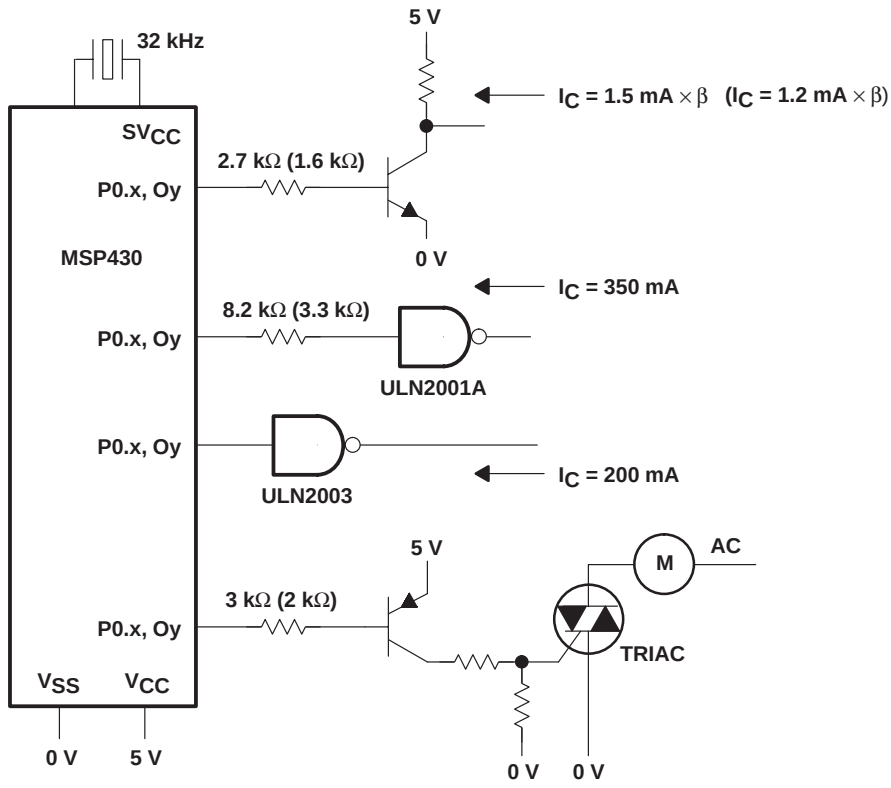


Figure 3–3. Output Buffering

3.1.4 Universal Timer/Port I/Os

If the Universal Timer/Port is not used for analog-to-digital conversion or is only partially used for this purpose, then the unused terminals are available as outputs that can be switched to high impedance. The Universal Timer/Port can be used in three different modes (see Figure 3–4):

- ❏ Two 8-bit timers, two inputs, one I/O, and 5 output terminals
- ❏ One 16-bit timer, two inputs, one I/O, and 5 output terminals
- ❏ An analog-to-digital converter with two to six output terminals

Ports TP0.0 to TP0.5 are completely independent of the analog-to-digital converter. Any of the ports can be used for the sensors and reference resistors.

After a power-up, the data register is set to zero and all TP0.x ports are switched to high impedance.

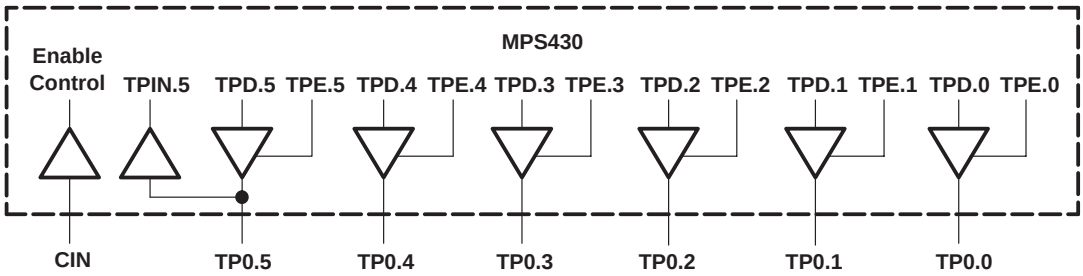


Figure 3–4. The I/O Section of the Universal Timer/Port Module

3.1.4.1 I/Os Used with the Analog-to-Digital Converter

The analog-to-digital conversion uses terminal CIN and at least two of the TP0.x terminals (one for the reference and one for the sensor to be measured); therefore up to 4 outputs are available. Bit instructions BIS.B, BIC.B, and XOR.B can only be used for the modification of the outputs. This is due to the location of the control bits in the data register TPD and data enable register TPE. The programming of the port is the same as described in the following section.

Note:

For precise ADC results, changes of the TP-ports during the measurement should be avoided. The board layout and the physical distance of the switched port determine the influence on the CIN terminal. Spikes coming from the switching of ports can change the result of a measurement. This is especially true if they occur near the crossing of the threshold voltage.

3.1.4.2 I/Os Used Without the ADC

This mode allows 5 outputs that can be switched to high impedance (TP0.0 to TP0.4) and one I/O terminal (TP0.5). Additionally, two 8-bit timers or one 16-bit timer are available. If one of the timers is used, only bit instructions BIT.B, BIS.B, BIC.B, or XOR.B can be used to operate the port. The four timer control bits are located in the data register TPD and data enable register TPE. If the MOV.B instruction is used, all the bits are affected.

Example 3–4. All Six Ports are Used as Outputs

All six ports are used as outputs. The possibilities of the port are shown in the following:

```
;
; Definitions for the Counter Port
;
TPD      .EQU     04Eh                    ; Data Register
TPE      .EQU     04Fh                    ; Data Enable Register. 1:
                                           output enabled
TP0      .EQU     001h                    ; TP0.0 bit address
TP1      .EQU     002h                    ; TP0.1 bit address
TP2      .EQU     004h                    ; TP0.2 bit address
TP3      .EQU     008h                    ; TP0.3 bit address
TP4      .EQU     010h                    ; TP0.4 bit address
TP5      .EQU     020h                    ; TP0.5 bit address
;
; Reset all ports and switch all to output direction
;
         BIC.B    #TP0+TP1+TP2+TP3+TP4+TP5,&TPD      ; Data to low
         BIS.B    #TP0+TP1+TP2+TP3+TP4+TP5,&TPE      ; Enable outputs
;
; Toggle TP0.0 and TP0.4, set TP0.5 and TP0.2 afterwards
;
         XOR.B    #TP0+TP4,&TPD                      ; Toggle TP0.0 and TP0.4
         BIS.B    #TP5+TP2,&TPD                      ; Set TP0.5 and TP0.2
;
; Switch TP0.1 and TP0.3 to HI–Z state
;
         BIC.B    #TP1+TP3,&TPE                      ; HI–Z state for TP0.1
                                                       and TP0.3
```

3.1.5 I/O Used for Fast Serial Transfers

The combination of hardware and software, shown in the following, allows a fast serial transfer with the MSP430 family. The data line needs to be Px.0. Any other port can be used for the clock line. Any data length is possible. The LSB is transferred first. This can be easily changed by using RLC instead of RRC.

```
;
P0OUT    .EQU     011h                    ; Port0 Output register
P0DIR    .EQU     012h                    ; Port0 Direction register
P00      .EQU     01h                     ; Bit address of P0.0: Data
P01      .EQU     02h                     ; Bit address of P0.1: Clock
;
         MOV      DATA,R5                 ; 1st 16bit data to R5
         CALL     #SERIAL_FAST_INIT       ; 1st transfer, initialization
         MOV      DATA1,R5                ; 2nd 16bit data to R5
         CALL     #SERIAL_FAST            ; 2nd transfer, LSB to MSB
         ....                             ; aso.
;
; Initialization of the fast serial transfer: uses SERIAL_FAST too
;
SERIAL_FAST_INIT                         ; Initialization part
         BIC.B    #P00+P01,&P0OUT         ; Reset P0.0 and P0.1
         BIS.B    #P00+P01,&P0DIR         ; P0.0 and P0.1 to output dir.
;
; Part for 2nd and all following transfers
;
SERIAL_FAST                      ; Initialization is made
         RRC      R5             ; LSB to carry           1 cycle
         ADDC.B   #P01,&P0OUT    ; Data out, set clock      4 cycles
         BIC.B    #P00+P01,&P0OUT ; Reset data and clock     5 cycles
;
         RRC      R5             ; LSB+1 to carry         1 cycle
         ADDC.B   #P01,&P0OUT    ; Data out, set clock      4 cycle
         BIC.B    #P00+P01,&P0OUT ; Reset data and clock     5 cycles
;
         ......                  ; Output all bits the same way
;
         RRC      R5             ; MSB to carry           1 cycle
         ADDC.B   #P01,&P0OUT    ; Data out, set clock      4 cycles
         BIC.B    #P00+P01,&P0OUT ; Reset data and clock     5 cycles
         RET
;
```

Each bit needs 10 cycles for the transfer, this results in a maximum baud rate for the transfer:

$$\text{Baud rate}_{max} = \frac{\text{MCLK}}{10}$$

This means if MCLK = 1.024 MHz then the maximum baud rate is 102.4 kbaud.

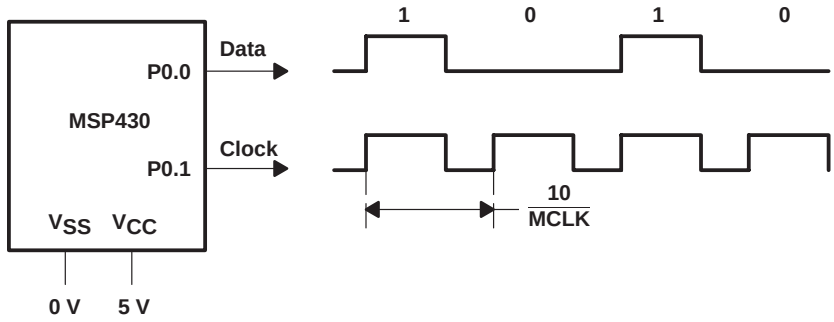


Figure 3–5. Connections for Fast Serial Transfer

3.2 Storage of Calibration Constants

Metering devices, such as electricity meters, gas meters etc., normally need to store calibration constants (offsets, slopes, limits, addresses, correction factors) for use during the measurements. Depending on the voltage supply (battery or ac), these calibration constants can be stored in the on-chip RAM or in an external EEPROM. Both methods are explained in the following sections.

3.2.1 External EPROM for Calibration Constants

The storage of calibration constants, energy values, meter numbers, and device versions in external EEPROMs may be necessary if the metering device is ac powered. This is because of the possibility of power failures.

The EEPROM is connected to the MSP430 by dedicated inputs and outputs. Three (or two) control lines are necessary for proper function:

- Data line SDA: an I/O port is needed for this bidirectional line. Data can be read from and written to the EEPROM on this line.
- Clock line SCL: any output line is sufficient for the clock line. This clock line can be used for other peripheral devices as long as no data is present on the data line during use.
- Supply line: if the current consumption of the idle EEPROM is too high, then switching of the EEPROM V_{CC} is needed. Three possible solutions are shown:
  - The EEPROM is connected to SV_{CC}. This is a very simple way to have the EEPROM powered off when not in use.
  - The EEPROM is switched on and off by an external PNP transistor driven by an output port.
  - The EEPROM is connected to 5 V permanently, when its power consumption is not a consideration.

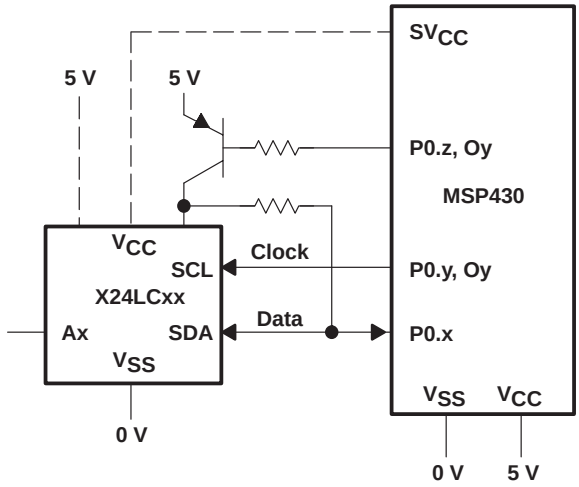


Figure 3–6. External EPROM Connections

An additional way to connect an EEPROM to the MSP430 is shown in Section 3.4, *I²C Bus Connection*, describing the I²C Bus.

Note:

The following example does not contain the necessary delay times between the setting and the resetting of the clock and the data bits. These delay times can be seen in the specifications of the EEPROM device. With a processor frequency of 1 MHz, each one of the control instructions needs 5 µs.

Example 3–5. External EEPROM Connections

The EEPROM, with the dedicated I/O lines, is controlled with normal I/O instructions. The SCL line is driven by O17, the SDA line is driven by P0.6. The line is driven high by a resistor and low by the output buffer.

```
P0OUT    .EQU     011h              ; Port0 Output register
P0DIR    .EQU     012h              ; Port0 Direction register
SCL      .EQU     0F0h              ; O17 controls SCL, 039h LCD Address
SDA      .EQU     040h              ; P0.6 CONTROLS SDA
LCDM     .EQU     030h              ; LCD control byte
;
; INITIALIZE I2C BUS PORTS:
; INPUT DIRECTION:   BUS LINE GETS HIGH
; OUTPUT BUFFER LOW: PREPARATION FOR LOW SIGNALS
;
```

```
        BIC.B    #SDA,&P0DIR              ; SDA TO INPUT DIRECTION
        BIS.B    #SCL,&LCDM+9             ; SET CLOCK HI
        BIC.B    #SDA,&P0OUT              ; SDA LOW IF OUTPUT
        ...
;
; START CONDITION: SCL AND SDA ARE HIGH, SDA IS SET LOW,
; AFTERWARDS SCL GOES LO
;
        BIS.B    #SDA,&P0DIR              ; SET SDA LO (SDA GETS OUTPUT)
        BIC.B    #SCL,&LCDM+9             ; SET CLOCK LO
;
; DATA TRANSFER: OUTPUT OF A ”1”
;
        BIC.B    #SDA,&P0DIR              ; SET SDA HI
        BIS.B    #SCL,&LCDM+9             ; SET CLOCK HI
        BIC.B    #SCL,&LCDM+9             ; SET CLOCK LO
;
; DATA TRANSFER: OUTPUT OF A ”0”
;
        BIS.B    #SDA,&P0DIR              ; SET SDA LO
        BIS.B    #SCL,&LCDM+9             ; SET CLOCK HI
        BIC.B    #SCL,&LCDM+9             ; SET CLOCK LO
;
; STOP CONDITION: SDA IS LOW, SCL IS HI,     SDA IS SET HI
;
        BIC.B    #SDA,&P0DIR              ; SET SDA HI
        BIS.B    #SCL,&LCDM+9             ; Set SCL HI
;
```

The examples, shown in the previous text, for the different conditions can be implemented into a subroutine, which outputs the contents of a register. This shortens the necessary ROM code significantly. Instead of line Ox for the SCL line another I/O port P0.x can be used. See Section 3.4, *I^2C Bus Connection*, for more details of such a subroutine.

3.2.2 Internal RAM for Calibration Constants

The internal RAM can be used for storage of the calibration constants, if a permanently connected battery is used for the power supply. The use of low power mode 3 or 4 is necessary for these kinds of applications and can get battery life times reaching 8 to 12 years.

3.3 M-Bus Connection

The MSP430 connection to the M-Bus (metering bus) is shown in Figure 3–7. Three supply modes are possible when used with the TSS721:

- ❏ Remote supply: The MSP430 is fully powered from the TSS721
- ❏ Remote supply/battery support: The MSP430 power is supplied normally from the TSS721. If this power source fails, a battery is used for backup power to the MSP430
- ❏ Battery Supply: The MSP430 is always supplied from a battery.

All these operating modes are described in detail in the *TSS721 M-Bus Transceiver Applications Book*.

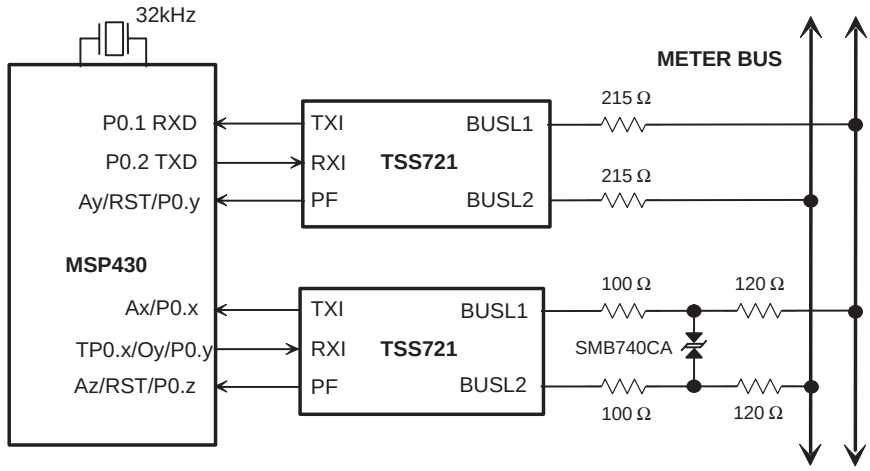


Figure 3–7. TSS721 Connections to the MSP430

Two different TSS721 connections are shown in Figure 3–7:

- ❏ If the 8-bit interval timer with its UART is used then the upper connection is necessary. TXI or TX are connected to RXD (P0.1) and RXI or RX is connected to TXD (P0.2).
- ❏ If a strictly software UART or an individual protocol is used, then any input and output combination can be used

The second connection uses a proven hardware for environments with strong EMV conditions. The 40-V suppressor diode gives the best results with this configuration.

For more details, see Section 3.8, *Power Supplies for MSP430 Systems*.

3.4 I2C Bus Connection

If more than one device is to be connected to the I2C-Bus, then two I/O ports are needed for the control of the I2C peripherals. This is needed to switch SDA and SCL to a high-impedance state.

Figure 3–8 shows the connection of three I2C peripherals to the MSP430:

- An EEPROM with 128x8-bit data
- An EEPROM with 2048x8-bit data
- An 8-bit DAC/ADC

The bus lines are driven high by the Rp resistors (P0.x is switched to input direction) and low by the output ports itself (P0.x is switched to output direction).

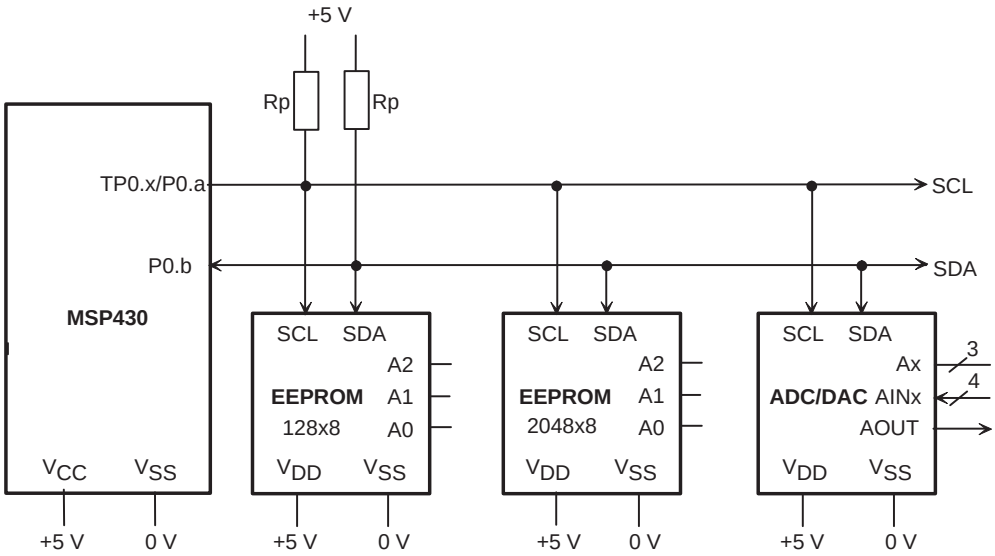


Figure 3–8. I2C Bus Connections

The following software example shows a complete I2C handler. It is designed for an EEPROM 24C65 with the following values:

- MCLK Frequency: 3.8 MHz
- Address Length: 13 bits
- Device Code: selectable by Code definition

```
; I2C–Handler: Transmission of 8–bit data via the I2C–bus
; Author: Christian Hernitscheck TID
;
; Definitions for I2C Bus
```

```
;
SCL      .EQU     040h                    ; P0.6 controls SCL line (pull–up)
SDA      .EQU     080h                    ; P0.7 controls SDA line (pull–up)
SCLIN    .EQU     010h                    ; P0 input register P0IN
SDAIN    .EQU     010h                    ; P0 input register P0IN
SCLDAT   .EQU     011h                    ; P0OUT register address
SDADAT   .EQU     011h                    ; P0 output direction register P0DIR
SCLEN    .EQU     012h                    ; P0DIR register address
SDAEN    .EQU     012h                    ; P0 direction register
Code     .equ     0A0h                    ; Device Code 10 (24C65)
;
Address  .EQU     0200h                   ; address pointer for EEPROM
I2CData  .EQU     0202h                   ; used for I2C protocoll
;
; Register definitions
;
Data     .EQU     R5
Count    .EQU     R6
Mask     .EQU     R7
;
; Initialization in main program
;
         BIC.B    #SCL+SDA,&SDAEN         ; SCL and SDA to input direction
         BIC.B    #SCL+SDA,&SDADAT        ; SCL and SDA output buffer lo
         ...                              ; Continue
;
; Subroutines of the I2C–Handler
;
; Write 8–bit data <data> into EEPROM address <address>:
;
; Call   MOV      <address>,Address       ; EEPROM data address
;        MOV.B    <data>,I2CData          ; 8–bit data
;        CALL     #I2C_Write              ; Call subroutine
;        JC       Error                   ; Acknowledge error
;        JN       Error                   ; Arbitration error
;        ...                              ; Continue program
```

```
;
; Read 8–bit data from EEPROM address <address>:
;
;        MOV      <address>,Address          ; EEPROM data address
;        CALL     #I2C_Read                  ; Call subroutine
;        JNC      Error                      ; Acknowledge error
;        JN       Error                      ; Arbitration error
;        ...                                 ; Data in I2CData (byte)
;
; Status Bits on return:
;
; C: Acknowledge Bit
; N: 1: Arbitration Error        0: no error
;
; Used Registers: R5 = Data  (pushed onto Stack)
;                                  R6 = Count (pushed onto Stack)
;                                  R7 = Mask  (pushed onto Stack)
;
; Used RAM:                        Address           0200h
;                                  I2CData           0202h
;                                  I2CData+1         0203h
;                                  I2CData+2         0204h
;                                  I2CData+3         0205h
;                                  I2CData+4         0206h
;
;
I2C_Write         MOV.B   I2CData,I2CData+3  ; Data to be written to EEPROM
         CALL     #ControlByte                ; Control byte
         MOV.B    Address+1,I2CData+1        ; Hi byte of EEPROM address
         AND.B    #01Fh,I2CData+1            ; Delete A2, A1 and A0 bits
         MOV.B    Address,I2CData+2          ; Lo byte of EEPROM address
         JMP      I2C                        ; To common part
;
I2C_Read CALL     #ControlByte               ; I2CData   = Control byte
         MOV.B    Address+1,I2CData+1        ; Hi byte of EEPROM address
         AND.B    #01Fh,I2CData+1            ; Delete A2, A1 and A0 bits
```

```
        MOV.B   Address,I2CData+2       ; Lo byte of EEPROM address
        MOV.B   I2CData,I2CData+4       ; Control byte 2
        BIS.B   #01h,I2CData+4           , To common part
;
; Common I2C–Handler
;
I2C     PUSH    Count                   ; Save registers
        PUSH    Data
        PUSH    Mask
        CLR     Count
        BIS.B   #SDA,&SDAEN             ; Start Condition: set SDA Lo
        MOV.B   I2CData,Data            ; Send slave address and RW bit
        CALL    #I2C_Send
        JC      I2C_Stop
;                                         Write or Read?
        BIT.B   #01h,I2CData+4          ;   _
        JC      I2C_SubRead             ; R/W bit is 1: read
;             _
; Write data (R/W = 0)
;
I2C_Data INC    Count
        CMP     #4,Count
        JEQ     I2C_Stop
        CALL    #I2C_Send
        JNC     I2C_Data
                                        ; Stop Condition:
I2C_Stop BIS.B  #SCL,&SCLEN             ; SCL = 'L'
        BIS.B   #SDA,&SDAEN             ; SDA = 'L'
        NOP                             ; Delay 7 cycles
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        BIC.B   #SCL,&SCLEN             ; SCL = 'H'
```

```
         CALL     #NOP9                    ; Delay 9 cycles
         BIC.B    #SDA,&SDAEN              ; SDA = 'H'
         CLRN                              ; Reset error flags
;
I2C_End  POP      Mask                     ; Restore registers
         POP      Data
         POP      Count
         RET                               ; Carry info valid
;             _
; Read data (R/W = 1)
;
I2C_SubRead INC   Count
         CMP      #3,Count
         JEQ      I2C_SubRead1
         CALL     #I2C_Send
         JC       I2C_Stop
         JMP      I2C_SubRead
I2C_SubRead1 BIS.B   #SCL,&SCLEN          ; SCL='L'
         CALL     #NOP9
         BIC.B    #SCL,&SCLEN              ; SCL='H'
         NOP
         NOP
         NOP
         NOP
         NOP                               ; Start condition:
         BIS.B    #SDA,&SDAEN              ; SCL='H', SDA='H' => 'L'
         MOV.B    I2CData+4,Data
         CALL     #I2C_Send                ; Send Control Byte
         JC       I2C_Stop
         BIS.B    #SCL,&SCLEN              ; SCL = 'L'
         BIC.B    #SDA,SDAEN               ; SDA = Input
         CLR      I2CData
         MOV      #8,Count                 ; Read 8 bits
;
I2C_Read1         BIC.B   #SCL,&SCLEN      ; SCL = 'H'
         BIT.B    #SDA,&SDAIN              ; Read data to carry
```

```
        RLC.B    I2CData                    ; Store received Bit
        NOP
        NOP
        BIS.B    #SCL,&SCLEN                ; SCL = 'L'
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        DEC      Count
        JNZ      I2C_Read1
;
        CALL     #I2C_Ackn                  ; Test acknowledge bit to C
        JMP      I2C_Stop
;
; Send byte
;
I2C_Send MOV.B   #80h,Mask                  ; Bit mask: MSB first
;
I2C_Send1        BIT.B   Mask,I2CData(Count)   ; Info bit –> Carry
        JC       I2C_Send2
        BIS.B    #SCL,&SCLEN                ; Info is 0: SCL = 'L'
        BIS.B    #SDA,&SDAEN                ; SDA = 'L'
        CALL     #NOP9
        BIC.B    #SCL,&SCLEN                ; SCL = 'H'
        JMP      I2C_Send3
I2C_Send2        BIS.B   #SCL,&SCLEN        ; Info is 1: SCL = 'L'
        BIC.B    #SDA,&SDAEN                ; SDA = 'H'
        CALL     #NOP9
        BIC.B    #SCL,&SCLEN                ; SCL = 'H'
        BIT.B    #SDA,SDAIN                 ; Arbitration
        JNC      Error_Arbit
;
I2C_Send3        CLRC
        RRC.B    Mask                       ; Next address bit
```

```
         NOP
         NOP
         NOP
         JNC      I2C_Send1                ; No Carry: continue
;
I2C_Ackn NOP
         NOP
         BIS.B    #SCL,&SCLEN              ; SCL = 'L' Acknowledge Bit
         BIC.B    #SDA,&SDAEN              ; SDA = 'H'
         CALL     #NOP8
         BIC.B    #SCL,&SCLEN              ; SCL = 'H'
         BIT.B    #SDA,&SDAIN              ; Read data to carry
         RET                               ; (acknowledge bit)
;
; I2C-Bus Error
Error_Arbit ADD   #2,SP                    ; Remove return address
         SETN                              ; Set arbitration error
         JMP      I2C_End
;
; Build control byte
;
ControlByte CLR   I2CData
         MOV.B    Address+1,I2CData        ; Hi byte of EEPROM address
         RRC      I2CData                   ; Shift MSBs to bits 3..1
         RRC      I2CData
         RRC      I2CData
         RRC      I2CData
         AND.B    #0Eh,I2CData             ; A2, A1 and A0
         ADD.B    #Code,I2CData            ; Add device code (24C65)
         RET
;
; Delay subroutine. Slows down I2C Bus speed to spec
;
NOP9     NOP                               ; 9 cycles delay
NOP8     RET                               ; 8 cycles delay
```

3.5 Hardware Optimization

The MSP430 permits the use of unused analog inputs (A7 to A0) and segment lines (S29 to S2) for inputs and outputs, respectively. The following two sections explain in detail how to program and use these inputs and outputs.

3.5.1 Use of Unused Analog Inputs

Unused analog-to-digital converter (ADC) inputs can be used as digital inputs or, with some restrictions, as digital outputs.

3.5.1.1 Analog Inputs Used for Digital Inputs

Any ADC input A7 to A0 can be used as a digital input. It only needs to be programmed (for example, during the initialization) for this function. Three things are important if this feature is used:

- Any activity at these digital inputs has to be stopped during ongoing sensitive ADC measurements. This activity will cause noise, which invalidates the ADC results. Activity in this case means:
  - No change of the AEN register (switching between digital and analog mode)
  - No input change at the digital ADC inputs (this rarely allows changing signals at these inputs).
- All bits that are switched to ADC inputs will read zero when read. Therefore, it is not necessary to clear them with software after reading.
- Not all analog inputs are implemented in a given device

Example 3–6. A0 – A4 are used as ADC Inputs and A5 – A7 as Digital Inputs

```
AIN      .EQU     0110h                    ; Address DIGITAL INPUT REGISTER
AEN      .EQU     0112h                    ; Address DIGITAL INPUT ENABLE REG.
A7EN     .EQU     080h                     ; Bits in Dig. Input Enable Reg.:
A6EN     .EQU     040h                     ; 0: ADC     1: Digital Input
A5EN     .EQU     020h                     ;
; INITIALIZATION: A7 TO A5 ARE SWITCHED TO DIGITAL INPUTS
; A4 TO A0 ARE USED AS ANALOG INPUTS
;
         MOV      #A7EN+A6EN+A5EN,&AEN     ; A7 TO A5 DIGITAL MODE
         ...
;
```

```
; NORMAL PROGRAM EXECUTION:
; CHECK IF A7 OR A5 ARE HIGH. IF YES: JUMP TO LABEL L$100
;
        BIT     #A7EN+A5EN,&AIN         ; A7 .OR. A5 HI?
        JNZ     L$100                   ; YES
                ...                     ; NO, CONTINUE
;
; CHECK IF ALL DIG. INPUTS A7 TO A5 ARE LOW. IF YES: Go to L$200
;
        TST     &AIN                    ; A7 TO A5 LO?
        JZ      L$200                   ; YES, (ANALOG INPUTS READ ZERO)
```

3.5.1.2 Analog Inputs Used as Digital Outputs

If outputs are needed then the unused ADC inputs with the current source connection can be used with the following restrictions:

- ❏ Only one ADC input can be high at a given time (1 out of n principle)
- ❏ Only the ADC inputs A0 to A3 are usable (only they are connected to the current source)
- ❏ The outputs can go high only while the ADC is not using the current source.
- ❏ The output current is directly related to the supply voltage, V_{CC}.
- ❏ The output voltage is only about 50% of the supply voltage, V_{CC}. Logic levels have to be carefully monitored. A transistor stage might be necessary (if not there already, e.g. for a relay).
- ❏ The output current is the current of the current source. Again, logic levels have to be carefully monitored. The pull-down resistor has to be big enough to allow the maximum output level.

The example in Figure 3–9 shows the ADC using inputs A0 and A1 as digital outputs driving two stages; a transistor stage (energy pulse, e.g. with an electricity meter) and a 3.3-V gate (3.3 V ensures that the input levels are sufficient).

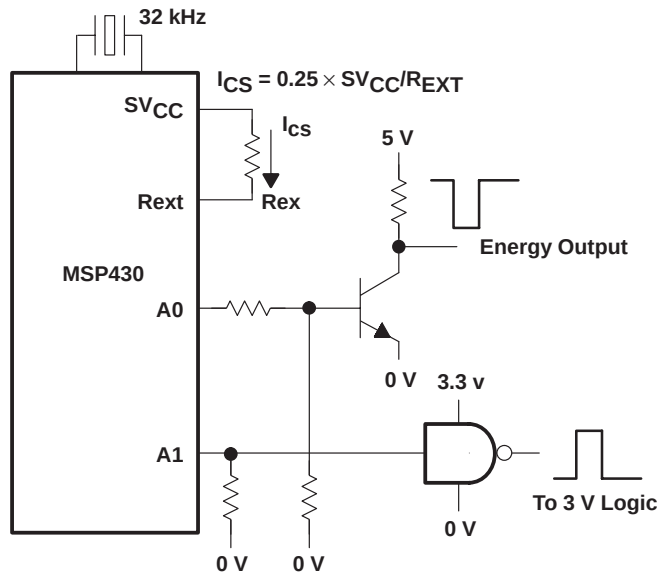


Figure 3–9. Unused ADC Inputs Used as Outputs

Example 3–7. Controlling Two Inputs as Outputs

To control the two outputs shown in Figure 3–9, the following software program is necessary:

```
ACTL     .EQU     0114h                    ; ADC CONTROL REGISTER ACTL
VREF     .EQU     02h                      ; 0: Ext. Reference    1: SVCC ON
A0       .EQU     0000h                    ; AD INPUT SELECT A0
A1       .EQU     0004h                    ;                                A1
CSA0     .EQU     0000h                    ; CURRENT SOURCE TO A0
CSA1     .EQU     0040h                    ;                                A1
CSOFF    .EQU     0100h                    ; CURRENT SOURCE OFF BIT
;
; SET A0 HI FOR 3ms: SELECT A0 FOR CURRENT SOURCE AND INPUT
         MOV      #VREF+A0+CSA0,&ACTL      ; PD = 0, SVCC = on
         CALL     #WAIT3MS                 ; WAIT 3ms
         BIS      #CSOFF,&ACTL             ; CURRENT SOURCE OFF;
;
; SET A1 HI FOR 3ms: SELECT A1 FOR CURRENT SOURCE AND INPUT
         MOV      #VREF+A1+CSA1,&ACTL      ; PD = 0, SVCC = on
         CALL     #WAIT3MS                 ; WAIT 3ms
```

```
BIS      #CSOFF,&ACTL                ; CURRENT SOURCE OFF
```

3.5.2 Use of Unused Segment Lines for Digital Outputs

The LCD driver of the MSP430 provides additional digital outputs, if the segment lines are not used. Up to 28 digital outputs are possible by the hardware design, but not all of them will be implemented on a given chip. The addressing scheme for the digital outputs O2 to O29 is as illustrated in Table 3–4.

Table 3–4 shows the dependence of the segment/output lines on the 3-bit value LCDP. When LCDP = 7, all the lines are switched to LCD Mode (segment lines). Only groups of four segment lines can be switched to digital output mode. LCDP is set to zero by the PUC (O6 to O29 are in use).

Note:

Table 3–4 shows the digit environment for a 4-MUX LCD display. The outputs O0 and O1 are not available: S0 and S1 are always implemented as LCD outputs. (digit 1).

The digital outputs Ox have to be addressed with all four bits. This means that 0h and 0Fh are to be used for the control of one output.

Only byte addressing is allowed for the addressing of the LCD controller bytes. Except for S0 and S1, the PUC switches the LCD outputs to the digital output mode (LCDP = 0).

Table 3–4. LCD and Output Configuration

Address	7 6 5 4	3 2 1 0	Digit Nr.	LCDP
03Fh	O29	O28	Digit 15	6 to 0
03Eh	O27	O26	Digit 14	6 to 0
03Dh	O25	O24	Digit 13	5 to 0
03Ch	O23	O22	Digit 12	5 to 0
03Bh	O21	O20	Digit 11	4 to 0 S20/S21
03Ah	O19	O18	Digit 10	4 to 0
039h	O17	O16	Digit 9	3 to 0
038h	O15	O14	Digit 8	3 to 0
037h	O13	O12	Digit 7	2 to 0
036h	O11	O10	Digit 6	2 to 0 S10/S11
035h	O09	O08	Digit 5	1 to 0
034h	O07	O06	Digit 4	1 to 0
033h	O05	O04	Digit 3	0
032h	O03	O02	Digit 2	0
031h	h g f e	d c b a	Digit 1	S0/S1

Example 3–8. S0 to S13 Drive a 4-MUX LCD

S0 to S13 drive a 4-MUX LCD (7 digits). O14 to O17 are set as digital outputs.

```
; LCD Driver definitions:
;
LCDM     .EQU     030h              ; ADDRESS LCD CONTROL BYTE
LCDM0    .EQU     001h              ; 0: LCD off      1: LCD on
LCDM1    .EQU     002h              ; 0: high         1: low Impedance
MUX      .EQU     004h              ; MUX: static, 2MUX, 3MUX, 4MUX
LCDP     .EQU     020h              ; Segment/Output Definition LCDM7/6/5
O14      .EQU     00Fh              ; O14 Control Definition
O15      .EQU     0F0h              ; O15
O16      .EQU     00Fh              ; O16
O17      .EQU     0F0h              ; O17
;
; INITIALIZATION: DISPLAY ON:       LCDM0 = 1
;                                   HI IMPEDANCE      LCDM1 = 0
;                                   4MUX:             LCDM4/3/2 = 7
;        O14 TO O17 ARE OUTPUTS:            LCDM7/6/5 = 3
;
         MOV.B    #(LCDP*3)+(MUX*7)+LCDM0,&LCDM      ; INIT LCD
         ...
;
; NORMAL PROGRAM EXECUTION:
; SOME EXAMPLES HOW TO MODIFY THE DIGITAL OUTPUTS O14 TO O17:
;
         BIS.B    #O14,&LCDM+8              ; SET O14, O15 UNCHANGED
         BIC.B    #O15+O14,&LCDM+8          ; RESET O14 AND O15
         MOV.B    #O15+O14,&LCDM+8          ; SET O14 AND O15
         MOV.B    #O17,&LCDM+9              ; RESET O16, SET O17
         XOR.B    #O17,&LCDM+9              ; TOGGLE O17, O16 STAYS UNCHANGED
```

3.6 Digital-to-Analog Converters

The MSP430 does not contain a digital-to-analog converter (DAC) on-chip, but it is relatively simple to implement the DAC function. Five different solutions with distinct hardware and software requirements are shown in the following:

- The R/2R method
- The weighted-resistors method
- Integrated DACs connected to the I^2C Bus
- Pulse width modulation (PWM) with the universal timer/port module
- Pulse width modulation with Timer A

3.6.1 R/2R Method

With a CMOS shift register or digital outputs, a DAC can be built for any bit length. The outputs Qx of the shift register switch the 2R-resistors to 0 V or V_{CC} according to the digital input. The voltage at the non-inverting input and also at the output voltage V_{out} of the operational amplifier is:

$$V_{out} = \frac{k}{2^n} \times V_{CC}$$

Where:

k	Value of the digital input word with n bits length
n	Number of Q outputs, maximum length of input word
V_{CC}	Supply voltage

Signed output is possible by level shifting or by splitting of the power supply ($+V_{CC}/2$ and $-V_{CC}/2$). With split power supplies the voltage at the output of the operational amplifier is:

$$Vout = \frac{k}{2^n} \times V_{CC} - \frac{V_{CC}}{2} = V_{CC}\left(\frac{k}{2^n} - \frac{1}{2}\right)$$

Advantages of the R/2R Method

- Only two different resistors are necessary (R and 2R)
- Absolute monotony over the complete output range
- Internal impedance independent of the digital value: impedance is always R
- Expandable to any bit length by the adding of shift registers
- With only three digital outputs (Ox, TP0.x, Portx), an inexpensive solution is possible.

If enough digital outputs are available in an application, then the shift register(s) can be omitted. The outputs QA to QH of Figure 3–10 are substituted by O outputs, ports or TP outputs of the MSP430.

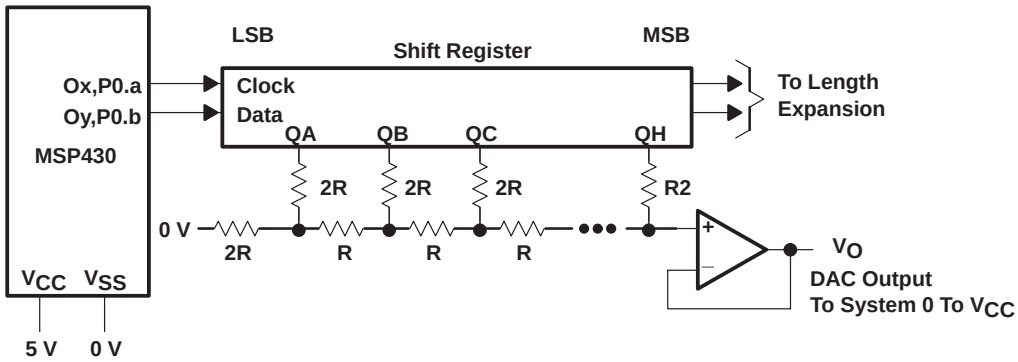


Figure 3–10. R/2R Method for Digital-to-Analog Conversion

3.6.2 Weighted Resistors Method

The simplest digital-to-analog conversion method, only (n+3) resistors and an operational amplifier are required for an n-bit DAC. This method is used when the DAC performance can be low.

The example shown in Figure 3–11 delivers 2^{n+1} different output voltage steps. They can be seen as signed if the voltage $V_{CC}/2$ is seen as a zero point. The output voltage Vout of this DAC is:

$$V_{out} = V_{ninv} - \sum I_n \times R = \frac{V_{CC}}{2} \times \left(1 + \left(a \times 2^{-1} + b \times 2^{-2} + c \times 2^{-3} \ldots + x \times 2^{-(n+1)}\right)\right)$$

Where:

V_{out}	Output voltage of the DAC
V_{ninv}	Voltage at the noninverting input of the operational amplifier ($V_{CC}/2$)
V_{CC}	Supply voltage of the MSP430 and periphery
R	Normalized resistor used with the DAC
a...x	Multiplication factors for the weighted resistors R to $2^n \times R$:

+1 if port is switched to V_{SS}
0 if port is switched to input direction (high impedance)
–1 if port is switched to V_{CC}

Normally all of the ports are switched to the same potential (VSS or V_{CC}) or are disabled. This allows signed output voltages referenced to $V_{CC}/2$.

- ❏ Advantage of the Weighted Resistor-Method: Simplicity
- ❏ Disadvantage: Monotony not possible due to resistor tolerances

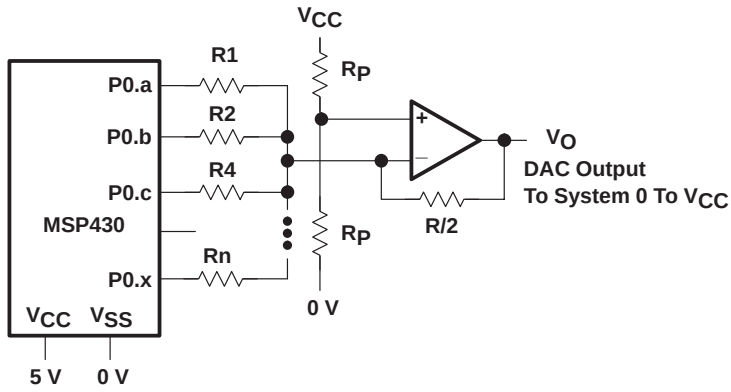


Figure 3–11. Weighted Resistors Method for Digital-to-Analog Conversion

3.6.3 Digital-to-Analog Converters Connected Via the I²C Bus

Figure 3–12 shows two different DACs that are connected to the MSP430 via the I²C Bus:

- ❏ A single output 8-bit DAC (with additional 4 ADC inputs); one analog output, AOUT, is provided.
- ❏ An octuple 6-bit DAC; eight analog outputs, DAC0 to DAC7, are available for the system

The generic software program to handle these devices is contained in the Section 3.4, *I²C Bus Connection*, explaining the I²C-Bus.

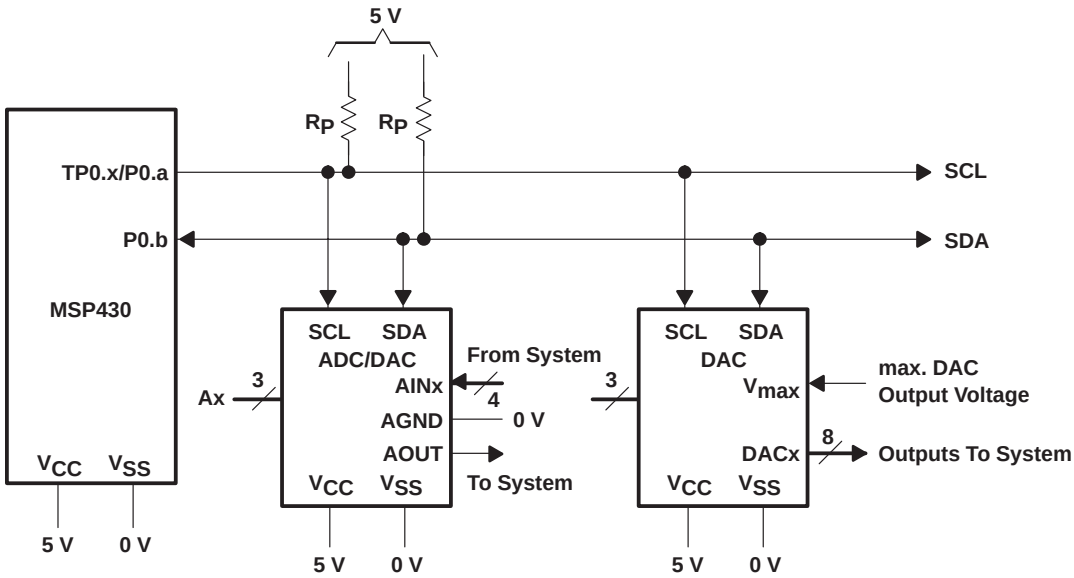


Figure 3–12. I²C-Bus for Digital-to-Analog Converter Connection

3.6.4 PWM DAC With the Universal Timer/Port Module

The two timers contained in the universal timer/port module can be used for one or two independent PWM generators. The ACLK frequency is used for the timing of these PWMs. The basic timer determines the period of the PWM signals. Its interrupt handler sets the programmed outputs and loads the two timer registers TPCNT2 and TPCNT1 with the negated pulse length values (see Table 3–5). The universal timer/port module terminates the pulses. Its interrupt handler resets the outputs when the counters, TPCNTx, overflow from 0FFh to 00h. The length of one step is always 1/ACLK, which is 30.51758 µs if a 32.768 kHz crystal is used.

Table 3–5 shows the necessary basic timer frequency , which is dependent on the PWM resolution used.

Table 3–5. Resolution of the PWM-DAC

Resolution Bits	Resolution Steps	Basic Timer Frequency
8	256	128
7	128	256
6	64	512
5	32	1024

Table 3–6 shows the values to be written into timer register TPCNT1 or TPCNT2 to get a certain PWM output value (related to V_{CC}); it is the desired value subtracted from the resolution value. The PWM switch (a byte in RAM) determines if the output is enabled (1) or disabled (0).

Table 3–6. Register Values for the PWM-DAC

PWM Output (Relative to V_{CC})	TPCNTx Value 256 Steps	TPCNTx Value 128 Steps	TPCNTx Value 64 Steps	TPCNTx Value 32 Steps	PWM Switch
0	x	x	x	x	0
0.25	C0h	E0h	F0h	F8h	1
0.50	80h	C0h	E0h	F0h	1
0.75	40h	A0h	D0h	E8h	1
1.00	00h	80h	C0h	E0h	1

Note:

The interrupt latency time plays an important role for this kind of PWM generation. Real time programming is necessary. Therefore, the first instruction of each interrupt handler must be the EINT instruction.

Example 3–9. PWM DAC With Timer/Port Module

Two PWM outputs with 8-bit resolution are realized. To get the highest speed, TP0.2 and TP0.1 are used as outputs (they have the same bit addresses as the flags RC2FG and RC1FG). The schematic is shown in Figure 3–13. The output ripple is shown in an exaggerated manner. If the PWM information is needed (as for DMC) then the signal at TP0.x is used directly.

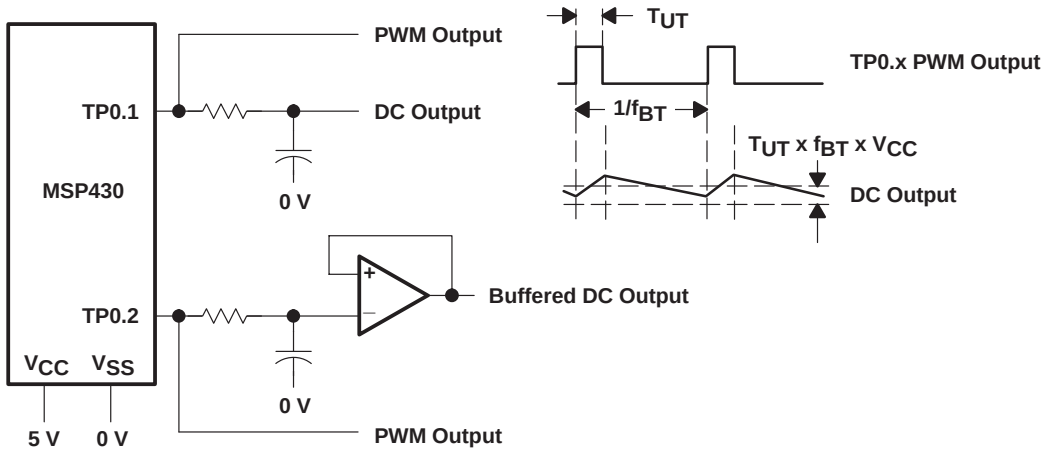


Figure 3–13. PWM for the DAC

Figure 3–14 illustrates the counting of the 8-bit counter during the PWM generation. The interrupt handler of the basic timer sets the 8-bit counter to a negative number of counts (–n1) and sets the output to high; the interrupt handler of the universal timer/port resets the output to zero when it overflows.

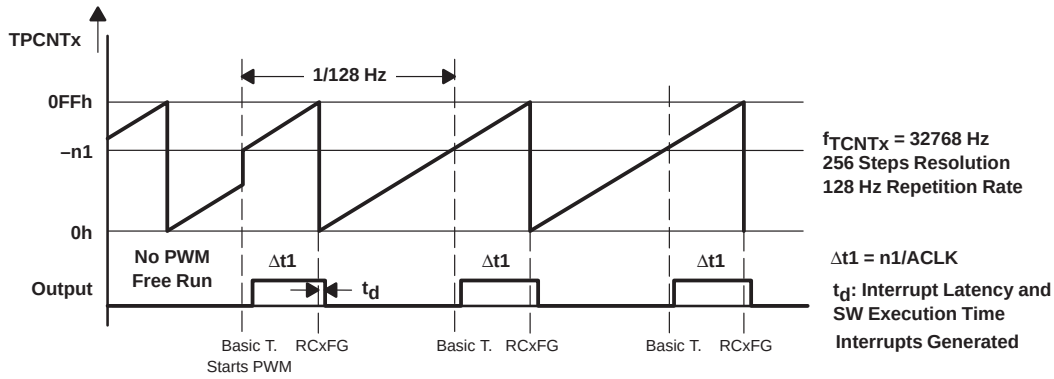


Figure 3–14. PWM Timing by the Universal Timer/Port Module and Basic Timer

```
; MSP430 Software for 8 bit PWM with Universal/Timer Port
; Definitions of the MSP430 hardware
;
Type     .equ     310             ; 310: MSP43C31x   0: others
BTCTL    .equ     040h            ; Basic Timer: Control Reg.
BTCNT1   .equ     046h            ;              Counter
BTCNT2   .equ     047h            ;              Counter
BTIE     .equ     080h            ;            : Intrpt Enable
SSEL     .equ     080h            ;
DIV      .equ     020h            ; BTCTL: xCLK/256
IP2      .equ     004h            ; BTCTL: Clock Divider2
IP1      .equ     002h            ;
IP0      .equ     001h            ;        Clock Divider0
;
SCFQCTL  .equ     052h            ; FLL Control Register
MOD      .equ     080h            ; Modulation Bit: 1 = off
;
CPUoff   .equ     010h            ; SR: CPU off bit
GIE      .equ     008h            ; SR: General Intrpt enable
;
```

```
TPCTL    .equ     04Bh            ; Timer Port:     Control Reg.
TPCNT1   .equ     04Ch            ;                 Counter Reg.Lo
TPCNT2   .equ     04Dh            ;                 Counter Reg.Hi
TPD      .equ     04Eh            ;                 Data Reg.
TPE      .equ     04Fh            ;                 Enable Reg.
TP1      .equ     002h            ; Bit address     TP0.1
TP2      .equ     004h            ;                 TP0.2
TP3      .equ     008h                              TP0.3
TP4      .equ     010h            ;                 TP0.4
TP5      .equ     020h            ;                 TP0.5
;
         .if      Type=310        ; MSP430C31x?
TPIE     .equ     004h            ; ADC: Intrpt Enable Bit
         .else
TPIE     .equ     008h            ; MSP430C32x configuration
         .endif
IE2      .equ     001h            ; Intrpt Enable Byte
TPSSEL3 .equ      080h            ;
TPSSEL2 .equ      040h
TPSSEL1 .equ      080h            ; Selects clock input (TPCTL)
TPSSEL0  .equ     040h            ;
ENB      .equ     020h            ; Selects clock gate (TPCTL)
ENA      .equ     010h            ;
EN1      .equ     008h            ; Gate for TPCNTx   (TPCTL)
RC2FG    .equ     004h            ; Carry of HI counter (TPCTL)
RC1FG    .equ     002h            ; Carry of LO counter (TPCTL)
EN1FG    .equ     001h            ; End of Conversion Flag ”
B16      .equ     080h            ; Use 16–bit counter (TPD)
; RAM Definitions
;
SW_PWM   .equ     0200h           ; Enable bits for TP0.2 and TP0.1
TIM_PWM1 .equ     0201h           ; Calc. PWM result PWM1
TIM_PWM2 .equ     0202h           ; Calc. PWM result PWM2
;
;=========================================================
;
```

```
        .sect   ”INIT”,0F000h           ; Initialization Section
;
INIT    MOV     #0300h,SP               ; Initialize Stack Pointer
        MOV.B   #IP2+IP1+IP0,&BTCTL     ; Basic Timer 128Hz
;
        MOV.B   #TPSSEL0+ENA,&TPCTL     ; ACLK, EN1=1, TPCNT1
        CLR.B   &TPCNT1                 ; Clear PWM regs
        CLR.B   &TPCNT2
        CLR.B   &TPD                    ; Output Data = Low
        MOV.B   #TPSSEL2+TP2+TP1,&TPE   ; TPCNT2: ACLK
        BIS.B   #TPIE+BTIE,&IE2         ; INTRPTS on
        CLR.B   SW_PWM                  ; No PWM output
        BIC.B   #RC2FG+RC1FG,&TPCTL     ; Reset flags
        EINT
        ...                             ; Continue with SW
;
; Start both PWMs: calculation results in R6 and R5
;
        MOV.B   R6,TIM_PWM1             ; (256 – result1)
        MOV.B   R5,TIM_PWM2             ; (256 – result2)
        BIS.B   #TP2+TP1,SW_PWM         ; Enable PWM2 and PWM1
        ...                             ; Continue
;
; Disable PWM2: Output zero
;
        BIC.B   #TP2,SW_PWM             ; Disable PWM2
        ...
;
; Interrupt Handler for the Basic Timer Interrupt: 128Hz
;
BT_INT  BIC.B   #RC2FG+RC1FG,&TPCTL     ; Clear flags
        MOV.B   TIM_PWM2,&TPCNT2        ; (256 – time2)
        MOV.B   TIM_PWM1,&TPCNT1        ; (256 – time1)
        BIS.B   SW_PWM,&TPD             ; Switch on enabled PWMs
        RETI
;
```

```
; End of Basic Timer Handler
;-------------------------------------------------------
; Interrupt Handler for the Universal Timer/Port Module
; For max. speed TP0.2 and TP0.1 are used (same bit locations
; as RC2FG and RC1FG). If other locations are used, RLA
; instructions have to be inserted after the flag clearing
;
UT_HNDL PUSH.B    &TPCTL                    ; INTRPT from where?
        AND       #RC2FG+RC1FG,0(SP)        ; Isolate flags
        BIC.B     @SP,&TPCTL                ; Clear set flag(s)
        BIC.B     @SP+,&TPD                 ; Reset actual I/O(s)
        RETI
;
; End of Universal Timer/Port Module Handler
;-------------------------------------------------------
;
        .sect     "INT_VECT",0FFE2h
        .WORD     BT_INT                    ; Basic Timer Vector
        .if       Type=310
        .sect     "INT_VEC1",0FFEAh         ; MSP430C31x
        .else
        .sect     "INT_VEC1",0FFE8h         ; Others
        .endif
        .WORD     UT_HNDL                   ; UTP Vector (31x)
        .sect     "INT_VEC2",0FFFEh
        .WORD     INIT                      ; Reset Vector
```

Example 3–10. PWM Outputs With 7-Bit Resolution

Two PWM outputs with 7-bit resolution are realized. TP0.4 and TP0.3 are used as PWM outputs (this makes shifting necessary). The schematic is shown in Figure 3–15. Due to the inverting filters at the PWM outputs, the outputs of the MSP430 are also inverted to compensate for this. The output ripple is shown

in an exaggerated manner. If the PWM information is needed (as for DMC) then the signal at TP0.x can be used directly.

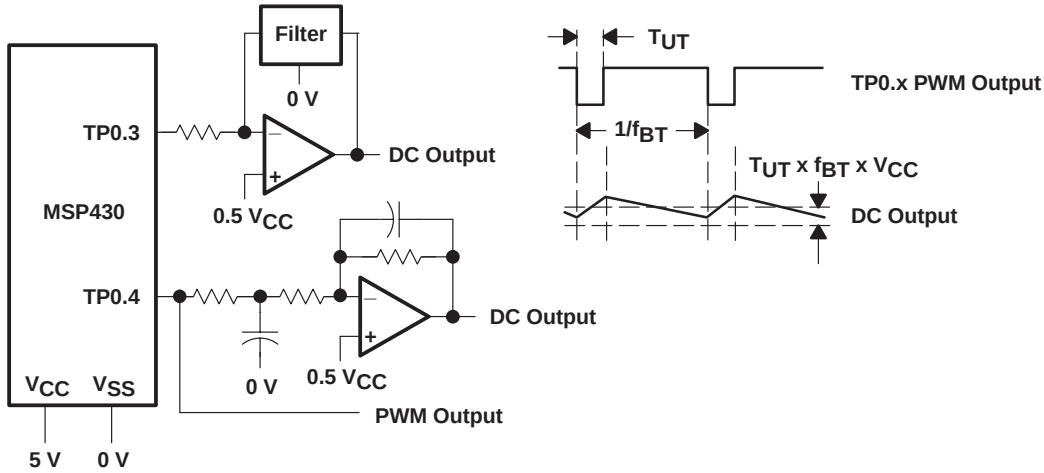


Figure 3–15. PWM for DAC

Figure 3–16 illustrates the operation of the 8-bit counter during the PWM generation. The interrupt handler of the basic timer sets the 8-bit counter to the negative number of counts (–n1) and resets the output to low; the interrupt handler of the universal timer/port sets the output to high when it overflows.

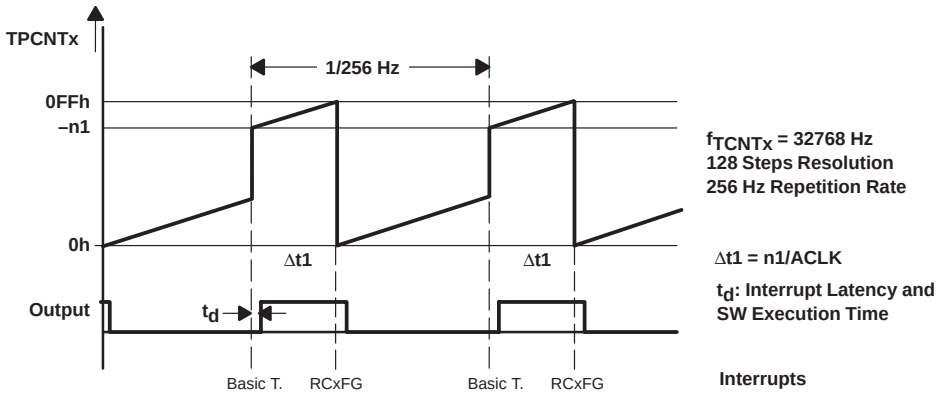


Figure 3–16. PWM Timing by Universal Timer/Port Module and Basic Timer

```
; MSP430 Software for 7 bit PWM with Universal/Timer Port
; Definitions of the MSP430 hardware like above
;
        .sect    ”INIT”,0F000h              ; Initialization Section
;
INIT    MOV      #0300h,SP                  ; Initialize SP
        MOV.B    #IP2+IP1,&BTCTL ; Basic Timer 256Hz
;
        MOV.B    #TPSSEL0+ENA,&TPCTL        ; ACLK, EN1=1, TPCNT1
        CLR.B    &TPCNT1                    ; Clear PWM regs
        CLR.B    &TPCNT2
        BIS.B    #TP4+TP3,&TPD              ; Output Data = high
        MOV.B    #TPSSEL2+TP2+TP1,&TPE      ; TPCNT2: ACLK
        BIS.B    #TPIE+BTIE,&IE2            ; INTRPTS on
        CLR.B    SW_PWM                     ; No output
        BIC.B    #RC2FG+RC1FG,&TPCTL        ; Clear flags
        EINT
        ...
; Start both PWMs: Calculation results in R6 and R5
;
        MOV.B    R6,TIM_PWM1                ; (128 – result)
        BIS.B    #TP3,SW_PWM                ; Enable PWM1
        MOV.B    R5,TIM_PWM2                ; (128 – result)
        BIS.B    #TP4,SW_PWM                ; Enable PWM2
```

```
         ...
; Disable PWMs: Output is zero
;
         BIC.B    #TP4+TP3,SW_PWM           ; No output
;
; Interrupt Handler for the Basic Timer Interrupt: 256Hz
; The enabled PWMs are switched on
;
BT_INT   BIC.B    #RC2FG+RC1FG,&TPCTL       ; Clear flags
         MOV.B    TIM_PWM2,&TPCNT2          ; (128 – time2)
         MOV.B    TIM_PWM1,&TPCNT1          ; (128 – time1)
         BIC.B    SW_PWM,&TPD               ; Switch on enabled PWMs
         RETI
;
; End of Basic Timer Handler
;-------------------------------------------------------------
; Interrupt Handler for the UT/PM. The PWM–channel that
; caused the interrupt is switched off.
;
UT_HNDL PUSH      R6                        ; Save R6
         MOV.B    &TPCTL,R6                 ; INTRPT from where?
         AND      #RC2FG+RC1FG,R6           ; Isolate flags
         BIC.B    R6,&TPCTL                 ; Clear set flag(s)
         RLA      R6                        ; To TP0.4/TP0.3
         RLA      R6
         BIS.B    R6,&TPD                   ; Set actual I/O(s)
         POP      R6                        ; Restore R6
         RETI
;
; End of Universal Timer/Port Module Handler
;-------------------------------------------------------------
; Vectors like with the example before
```

3.6.5 PWM DAC With the Timer_A

Timer_A of the MSP430 family is ideally suited for the generation of PWM signals. The output unit of each one of the (up to five) capture/compare registers is able to generate seven different output modes. The PWM generation depends mainly on which mode of the Timer_A was used.

- Continuous Mode: the timer register runs continuously upwards and rolls over to zero after the value 0FFFFh. The capture/compare register 0 is used like the other capture/compare registers. This mode allows up to five independent timings. The continuous mode is not intended for PWM applications. But, it can be used for relatively slow PWM applications, if other timings are also needed. Interrupt is used for the setting and the resetting of the PWM output. The output unit controls the PWM output and the interrupt handler adds the next time interval to the capture/compare register and modifies the mode of the output unit (set, toggle, or reset).
- Up Mode: The timer register counts up to the content of capture/compare register 0 (here the period register) and restarts at zero when it reaches this value The capture/compare register 0 contains the period information for all other capture/compare registers.
- Up-Down Mode: The timer register counts up to the content of capture/compare register 0 (here the period register) and counts down to zero when it reaches this value. When zero is reached again, the timer register counts up again. capture/compare register 0 contains the period information for all other capture/compare registers.

All three modes are explained in detail in the Section 6.3, *Timer_A*. Software program examples are also given. If dc output is needed, the same output filters can be used as shown in the previous section. The only difference is the possible speed of the Timer_A (input frequency can be up to the MCLK frequency).

3.6.5.1 PWM DAC With Timer_A Running in Continuous Mode

Up to five completely different PWM generations are possible. If the Timer Register equals one of the four capture/compare latches (programmed to compare mode), the hardware task programmed to the output unit is performed (set, reset, toggle etc.) and an interrupt is requested. Figure 3–17 illustrates the generation of a PWM signal with the capture/compare registers 0. The interrupt handler is responsible for the following tasks:

- The time difference (represented by the clock count nx) to the next interrupt is added to the used capture/compare register by software: once $\Delta t0$, once $\Delta t1$

- ❏ The output unit is programmed to the appropriate mode: set TA0 if ∆t1 is added, reset TA0 if ∆t0 is added.
- ❏ Other tasks if necessary

Note:

The continuous mode is not the normal mode for PWM generation due to the software overhead that is necessary. It is used for this purpose only if other independent timings are necessary that cannot be realized with the up mode or the up-down mode.

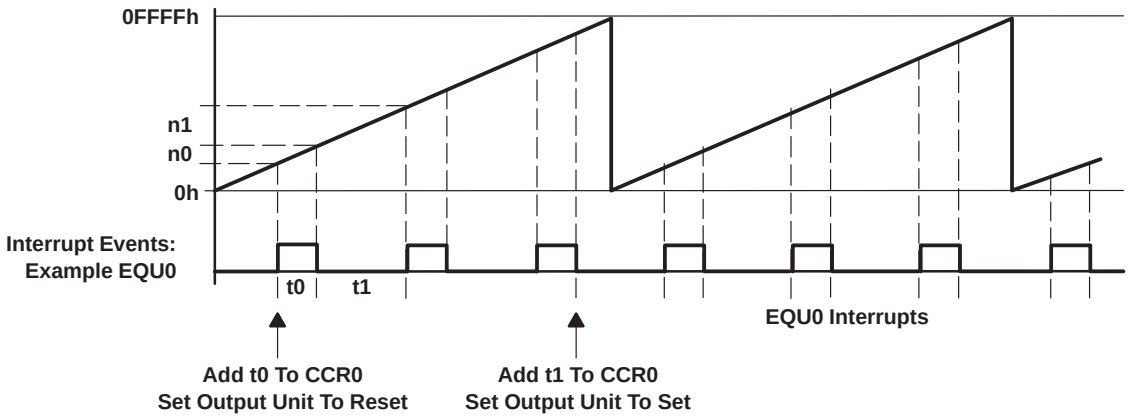


Figure 3–17. PWM Generation with Continuous Mode

3.6.5.2 PWM DAC With Timer_A Running in Up Mode

Up to four different PWM generations with an equal period (repetition rate) are possible. If the timer register equals one of the four capture/compare latches (programmed to compare mode), the hardware task programmed to the output unit is performed (set, reset, toggle etc.) and an interrupt is requested. During the execution of the interrupt handler, the necessary software task is completed. No reloading of the capture/compare register is necessary except if the pulse width changes. If the timer register reaches the programmed value of the capture/compare register 0, then it is reset to zero and restarts there. Figure 3–18 illustrates the generation of two independent PWM signals with the capture/compare registers 1 and 2.

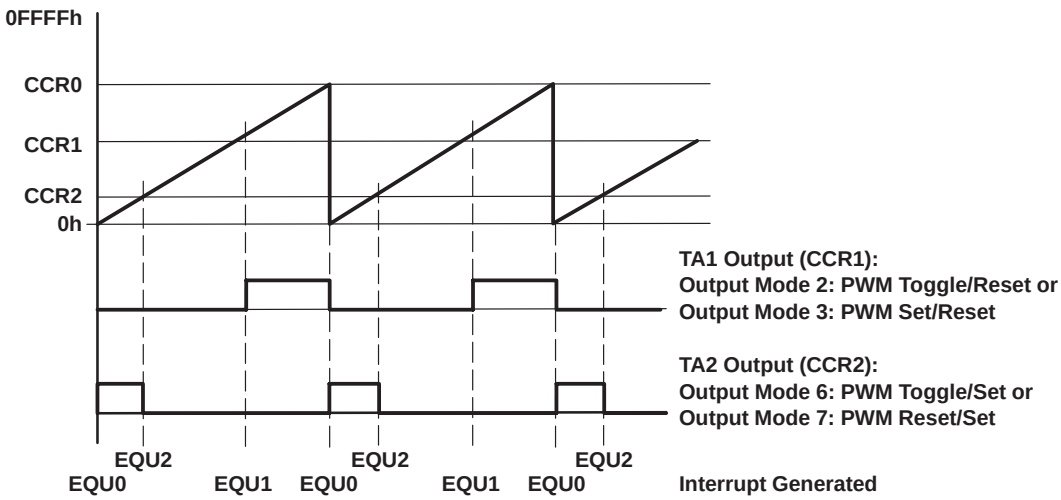


Figure 3–18. PWM Generation With Up Mode

3.6.5.3 PWM-DAC With Timer_A Running in Up-Down Mode

Up to four different PWM generations with an equal period are possible. If the timer register equals one of the four capture/compare latches (programmed to compare mode), the hardware task programmed to the output unit is performed (set, reset, toggle etc.) and an interrupt is requested. During the interrupt handler, the necessary software task is completed. No reloading of the capture/compare register is necessary except if the pulse width changes. The timer register continues to count upward until the value of capture/compare register 0 is reached. Then it counts downward to zero. When it reaches the value of a capture/compare register, the programmed task is made by the output unit and an interrupt is requested again. When zero is reached, the sequence restarts. This way, symmetric PWM generation is possible. The value of the capture/compare register is reached twice for each up-down cycle. Figure 3–19 illustrates the generation of two independent PWM signals with the capture/compare registers 1 and 3.

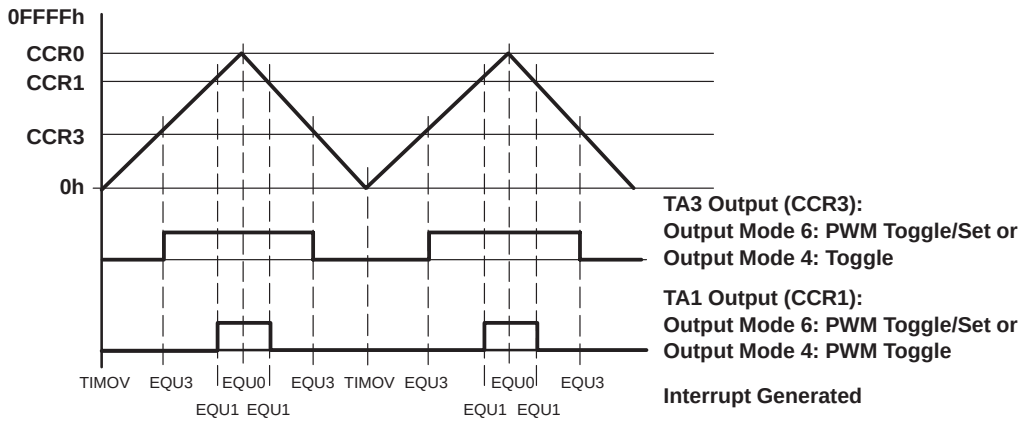


Figure 3–19. PWM Generation with Up-Down Mode

3.7 Connection of Large External Memories

For a lot of MSP430 applications, it is necessary to be able to store large amounts of measured data. For this purpose external memories can be used:

- ❏ Dynamic RAMs like the TMS44460 (1M × 4 bits)
- ❏ Synchronous Dynamic RAMs like the TMS626402 (2M × 4 bits)
- ❏ Flash memories like the TMS28F512A (512K × 8-bits)
- ❏ EEPROMs

DRAM versions with a self-refresh feature are recommended, otherwise the necessary refresh cycles would waste too much of the processing time.

Figure 3–20 shows the simplest way to control external memory. The unused LCD segment lines are used for addressing and control of the external memory. Four bidirectional I/O lines of port 0 (or another available port) are used for the bidirectional exchange of data. The necessary steps to read from or write to the example TMS44460 DRAM memory are:

1) Output row address to address lines A9 to A0
2) Set the RAS control line low
3) Output column address to address lines A9 to A0
4) Set CAS control lines low and reset them back to high
5) If a read is desired, set OE low and W control lines high. Then read data from DQ4 to DQ1.
6) If a write is desired, set OE high, set W low, and then write the data to DQ4 to DQ1.

The proposal shown in Figure 3–20 needs approximately 200 MCLK cycles for each block of 4-bit nibbles when the O-output lines are used.

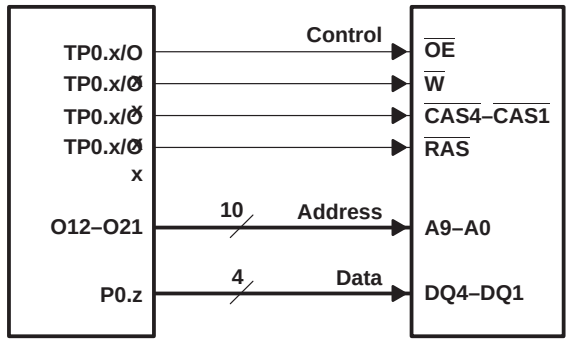


Figure 3–20. External Memory Control With MSP430 Ports

Example 3–11. External Memory Connected to the Outputs

For the circuit shown in Figure 3–20, the 10 address lines of an external memory are connected to the O-outputs, O12 (LSB) to O21 (MSB). The subroutine O_HNDLR is used for the row and column addressing. The driver software and the subroutine call follows:

```
N         .EQU     10/2              ; 10 O–outputs are controlled (O13 to O4)
O_STRT    .EQU     037h              ; Control byte for O12 and O13 (1st byte)
;
          MOV      #03FFh,R5         ; Start with row addressing
          CALL     #O_HNDLR
          ...                        ; Output @RAS signal
          MOV      R9,R5             ; Column address in R9
          CALL     #O_HNDLR          ; Output column address
          ...                        ; Output @CAS signals
;
; Subroutine outputs address info in R5 to O–outputs
; Bit 0 is written to the MSB of the O–outputs. R5 is destroyed
; Execution time:  69 cycles for  8 O–outputs (including CALL)
;                 129 cycles for 16 O–outputs (like above)
;
O_HNDLR   CLR      R6                ; Clear counter
O_HN      MOV      R5,R4             ; Copy actual info
```

```
        AND     #3,R4                   ; Isolate next two address bits
        MOV.B   TAB(R4),O_STRT(R6)      ; Write address bits
        RRA     R5                      ; Prepare next two address bits
        RRA     R5
        INC     R6                      ; Increment counter
        CMP     #N,R6                   ; Through?
        JNZ     O_HN                    ; No, next two bits
        RET
;
; Table contains bit pattern used for the O–outputs
;
TAB     .BYTE   0,0Fh,0F0h,0FFh         ; Patterns 00, 01, 10, 11
```

Figure 3–21 gives an example to use when the LCD segment lines are not available. Two 8-bit shift registers are used for addressing and control of the external memory. Four bidirectional I/O lines of port 0 (or another available port) are used for the exchange of data. Instead of outputting the address and control signals in parallel, this solution's signals are output in series. The output enable signals G2 and G1 are used to omit bad signals that are due to the shifting of the information. The example shown in Figure 3–21 needs approximately 500 cycles for each block of 4-bit nibbles.

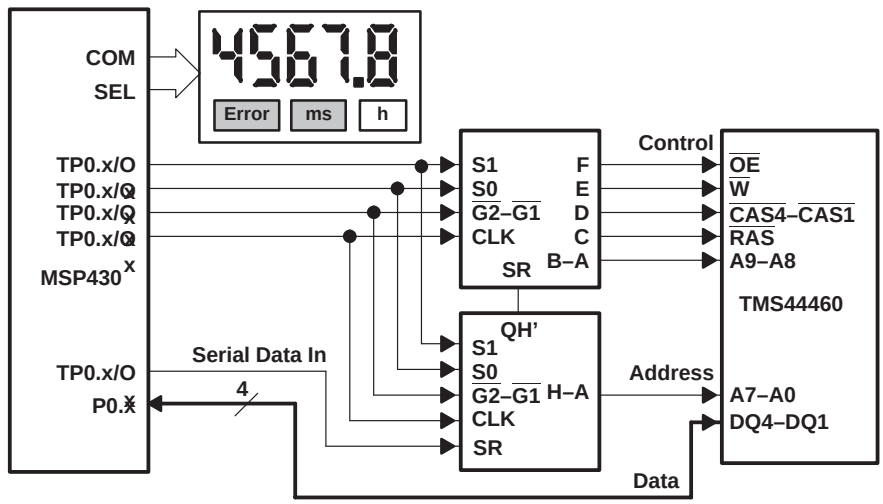


Figure 3–21. External Memory Control With Shift Registers

With nearly the same two hardware solutions, other external memories can be controlled also.

- Synchronous Dynamic RAM (TMS626402 2M × 4 bits) with 12 address lines and 6 control lines. Row and column addressing is used. It also uses 4 data bits.
- Flash memory (TMS28F512A 512K × 8-bit) with 16 address lines and 3 control lines. Direct addressing is used. It also uses 8 data bits.

Any combination of unused outputs (port, TP0.x, Oy) and shift registers can be used. If DRAMs without self-refresh are used, the low address bits should be controlled by a complete port (port 1 ,2, 3, or 4) to get minimum overhead for the refresh task.

The different versions of the MSP430C33x allow a much simpler and faster solution because of the five available I/O ports. Figure 3–22 illustrates the connection of an AT29LV010A EEPROM (128K × 8 bit) to the MSP430C33x. The example shown in Figure 3–22 needs approximately 30 to 50 MCLK cycles for each byte read or written. The control lines at the MSP430 are I/Os with no second function. All the peripheral functions are available and can be used freely. The MSP30C31x and 32x can address this type of memory by its TP0.x and Ox ports.

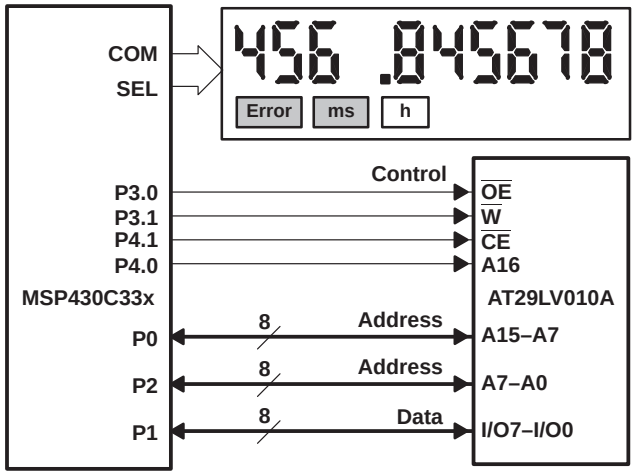


Figure 3–22. EEPROM Control With Direct Addressing by I/O Ports

Figure 3–23 shows the use of an MSP430C33x for the addressing of an external 1-MB RAM. The actual address of the external memory is stored in I/O Ports P0, P2, and P4. The special architecture of the MSP430 allows this method to be used. This method results in the fastest possible access time. The software used for addressing and reading of the next byte is in the following text (this assumes that the address ports are initialized).

```
;                                                                    Cycles
        INC.B   &P2OUT                  ; Address next Byte           4
        JNC     L$1                     ; No carry to A15..A8         2
        ADC.B   &P0OUT                  ; Carry to A15..A8            4
L$1     MOV.B   &P1IN,R15               ; Read data at Port1          3
```

The reading of a byte needs (4 + 2 + 3) = 9 cycles. An MCLK frequency of 3.8 MHz results in a read time of 2.37 µs. This access time can be compared with the internal access time of an 8-bit microcomputer. The initialization of a 64-KB memory block is shown in the following text (memory block 1).

```
;                                                                    Cycles
        MOV.B   #0,&P2OUT               ; A7..A0 = 00                 4
        MOV.B   #0,&P0OUT               ; A15..A8 = 00                4
        MOV.B   #CS1+WE,&P4OUT          ; Address memory block1       4
```

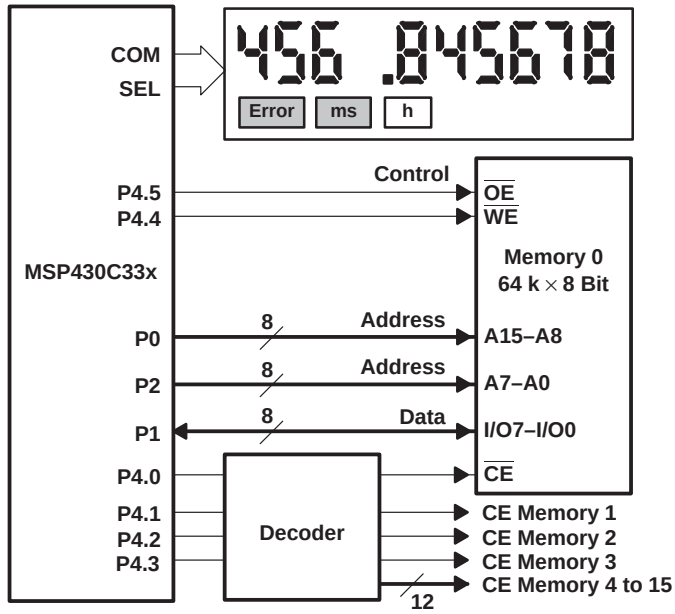


Figure 3–23. Addressing of 1-MB RAM With the MSP430C33x

Figure 3–24 shows how to address an external 1-MB RAM with an MSP430C31x. The actual address of the external memory (stored in the internal RAM) is output with the O-outputs (alternative use of the select lines) and the 6 TP ports. The software for addressing and reading of the next bytes is given in the following text (the address ports are initialized).

```
;                                                                    Cycles
        INC.B   A7_0                    ; Address next byte           4
        JNC     L$1                     ; No carry to A15..A8         2
        ADC.B   A15_8                   ; Carry to A15..A8            4
;
; Address A15 to A8 is output to O17 to O10.
; This part is necessary for only 0.4% of all accesses
;
        MOV.B   A15_8,R14               ; A15..8 –> R14               3
        MOV     R14,R15                 ;                             1
        AND     #3,R15                  ; Next 2 address bits         2
        MOV.B   TAB(R15),&036h          ; A9..8 to O11..10            6
        RRA     R14                     ; Next 2 address bits         1
        RRA     R14                                                   1
        MOV     R14,R15                                               1
        AND     #3,R15                  ; Address A7..6 aso.          2
        ...                             ; 4 x the same A15..6         36
;
; Address bits A7 to A0 output to TP–Port and O27/26
;
L$1     MOV.B   A7_0,&TPD               ; A7..A2 to TP–Port           6
        MOV.B   A7_0,R15                ; A1..A0 generated            3
        AND     #3,R15                  ; A1..A0 in R15               2
        MOV.B   TAB(R15),&LCDx          ; A1..A0 to O27 and O26       6
        MOV.B   &P0IN,R15               ; Read data at Port0          3
        ...                             ; Process data
;
TAB     .BYTE   0, 0Fh, 0F0h, 0FFh      ; For O–outputs
```

The reading of one byte needs 26 cycles (addresses A15 – A8 are unchanged) or 83 cycles when A15 – A8 must be changed. An MCLK frequency of 3.8 MHz results in 6.9 µs or 21.8 µs for one byte, respectively.

The decoding of the 64-KB memory blocks is made with a normal 4-to-16 line decoder.

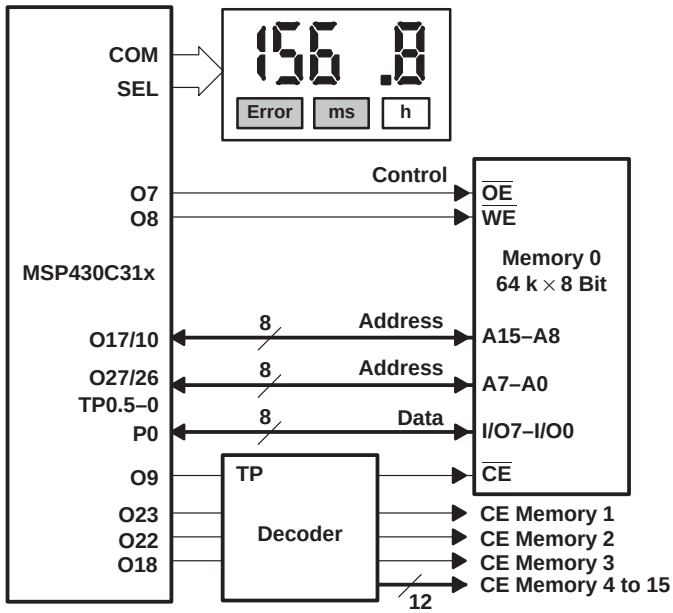


Figure 3–24. Addressing of 1-MB RAM With the MSP430C31x

3.8 Power Supplies for MSP430 Systems

There are various ways to generate the supply voltage(s) for the MSP430 systems. Due to the extremely low-power consumption of the MSP430 family, this is possible with batteries, accumulators, the M-Bus, fiber-optic lines, and ac. Every method uses completely different hardware and is explained in depth. Wherever possible, the formulas necessary for the hardware design are given too.

3.8.1 Battery-Power Systems

Due to the extremely low current consumption of the MSP430 family it is possible to run an MSP430 system with a 0.5-Ah battery more than 10 years. This makes possible applications that were impossible before. To reach such extended time spans, it is only necessary to observe some simple rules. The most important one is to always switch off the CPU when its not needed (e.g., after the calculations are completed). This reduces the current consumption from an operational low of 400 μA down to 1.6 μA.

The Figures 3–25 and 3–26 are drawn in a way that makes it easier to see how the battery needs to be connected to get the highest accuracy out of the ADC.

Figure 3–25 illustrates the MSP430C32x with its separated digital and analog supply terminals. This provides a separation of the noise-generating digital parts and the noise-sensitive analog parts.

Figure 3–26 shows how to best separate the two parts for the MSP430 family members with common supply terminals for the analog and digital parts of the chip.

If the battery used has a high internal resistance, RI, (like some long-life batteries) then the parallel capacitor Cch must have a minimum capacity. The supply current for the measurement part (which cannot be delivered by the battery) is delivered via Cch. The equation includes the small current coming from the battery.

$$C_{chmin} \geq t_{meas} \times \left(\frac{I_{AM}}{\Delta V_{ch}} - \frac{1}{R_I} \right)$$

Between two measurements, the capacitor Cch needs time, tch, to get charged-up to V_{CC} for the next measurement. During this charge-up time, the MSP430 system runs in low-power mode 3 to have the lowest possible power consumption. The charge-up time, tch, to charge Cch to 99% of V_{CC} is:

$$t_{chmin} \geq 5 \times C_{chmax} \times R_{imax}$$

Where:

I_{AM}	Medium system current (MSP430 and peripherals)	(A)
t_{meas}	Discharge time of Cch during measurement	(s)
ΔV_{ch}	Tolerable discharge of Cch during time tmeas	(V)
R_i	Internal resistance of the battery	(Ω)

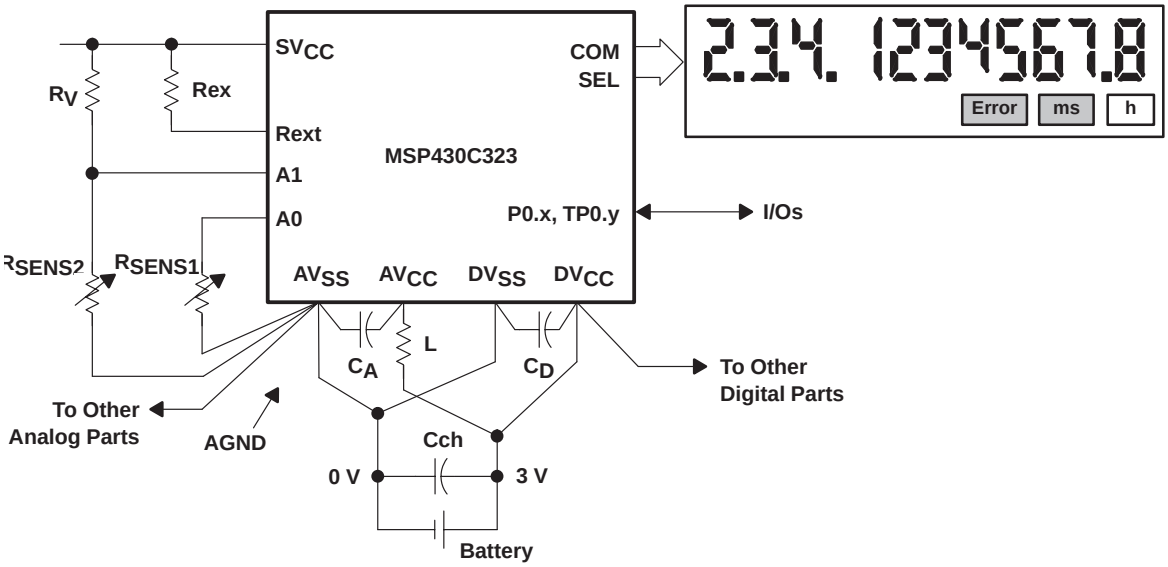


Figure 3–25. Battery-Power MSP430C32x System

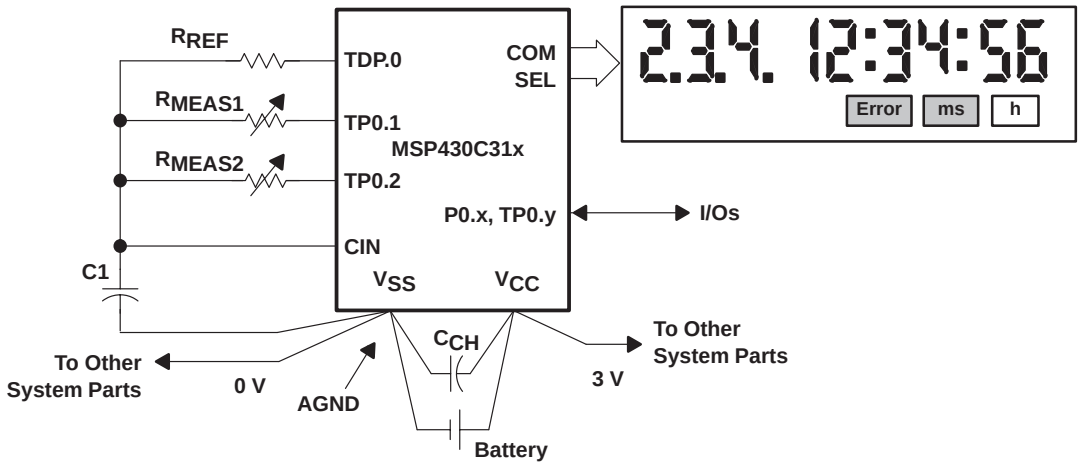


Figure 3–26. Battery-Power MSP430C31x System

Note:

The way the battery is connected to the MSP430 (shown in Figures 3–25 and 3–26) is not restricted to battery-driven MSP430 systems. The decoupling of the analog and the digital parts is necessary for all methods of supplied power. The following schematics are drawn in a simpler way to give better readability.

3.8.2 Accumulator-Driven Systems

The MSP430 can also be supplied from an accumulator. An advantage of this solution is that the MSP430 can also take over the battery management for the accumulator.

- Current Measurement: Summing up of the charge and discharge currents. If these currents (measured with sign) are multiplied with constants that are unique for the accumulator type used (e.g. NiCd, Pb) then it is possible to have a relatively accurate value for the actual charge. The current is measured with a shunt. The measured voltage drop is shifted into the middle of the ADC range by the current Ics (generated by the MSP430's internal current source) that flows through Rc. This method allows signed current measurements.

- Temperature Measurement: All of the internal processes of an accumulator (e.g., maximum charge, self discharge) are strongly dependent on the temperature of the pack. Therefore, the temperature of the pack is measured with a sensor and used afterwards with the calculations. When the MSP430's current source is used, the voltage drop of its current Ics across the sensor resistance is measured with the ADC input A2.

- Voltage Measurement: The voltage of an accumulator pack is an indication of the states full charge and complete discharge. Therefore, the voltage of the pack is measured with the voltage divider consisting of R1 and R2.

- Charge Control: Dependent on the result of the charge calculations, the MSP430 can decide if the charge transistor needs to be switched on or off. This decision can also be made in PWM (Pulse Width Modulation) mode. Figure 3–27 shows three possible charge modes. If replaceable accumulators are used, the charge control is not needed.

- Rest Mode Handling: During periods of non-use, the low power mode 3 of the MSP430 allows the control of the rest mode. The rest mode has nearly no current consumption. In fact, the supply current has the same magnitude as the self-discharge current of the accumulator. All system peripherals are switched off; the MSP430 wakes-up at regular intervals, which are controlled by its basic timer. It then calculates, every few hours, the amount of self discharge of the accumulator. This calculated value is subtracted from the actual charge level.

Figure 3–27 illustrates an MSP430 system driven by an accumulator. The battery management is done by the MSP430 also. The hardware needed is simple. As shown in the figure, just a few resistors and a temperature sensor. The actual charge of the accumulator is indicated in the LCD with a bar graph ranging from Empty to Full.

All necessary constants and a security copy of the actual charge are contained in an external EEPROM typically with 128 x 8 bits.

Note:

The hardware shown in Figure 3–27 can also be used for an intelligent accumulator controller. Only the hardware necessary for this task is shown. The measurement parts for voltage, current, and temperature are exactly the same as shown.

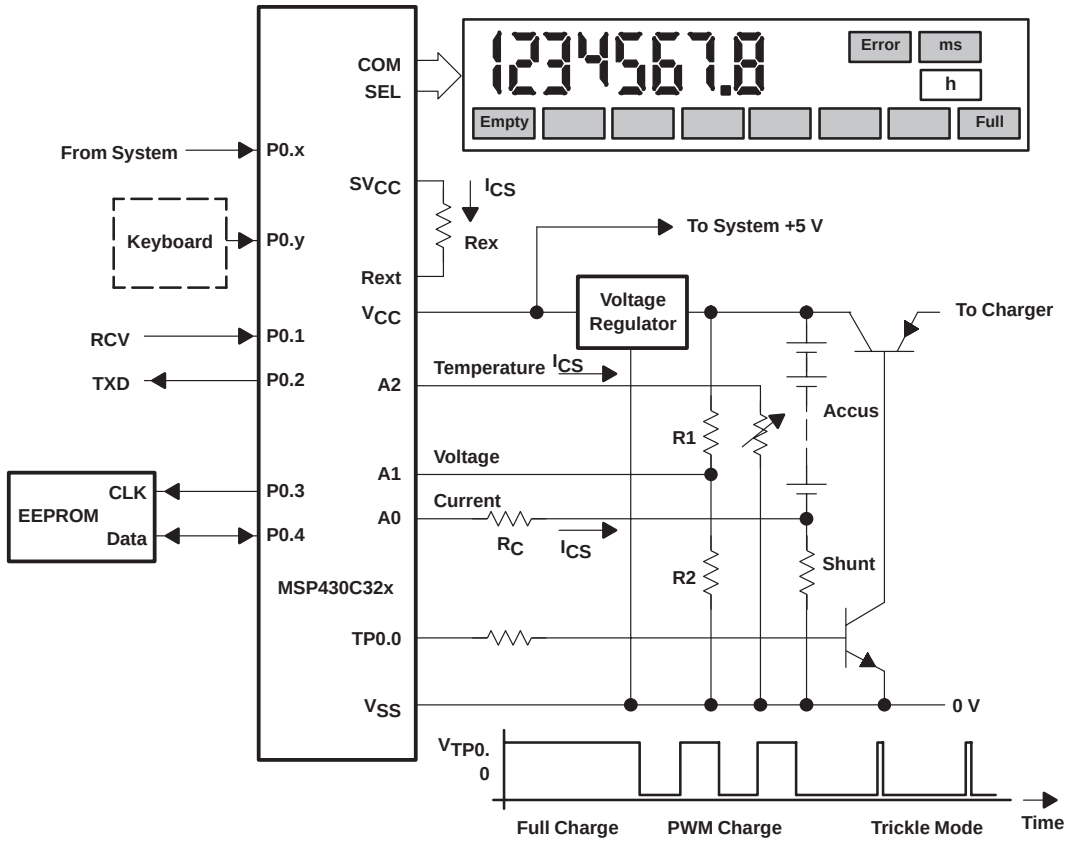


Figure 3–27. Accumulator-Driven MSP430 System With Battery Management

3.8.3 AC-Driven Systems

The current consumption of microcomputer systems gets more and more important for ac-driven systems. The lower the power consumption of a microcomputer system, the simpler and cheaper the power supply can be

3.8.3.1 Transformer Power Supplies

Transformers have two big advantages:

- ❏ Complete isolation from ac. This is an important security attribute for most systems.
- ❏ Very good adaptation to the needed supply voltage. This results in a good power efficiency.

Most ac-driven applications are only possible because of the isolation from the ac the transformer provides.

Half-Wave Rectification

Half-wave rectification uses only one half-wave of the transformer's secondary voltage, V_{SEC}, for the powering of an application. Figure 3–28 illustrates the voltages used with the equations.

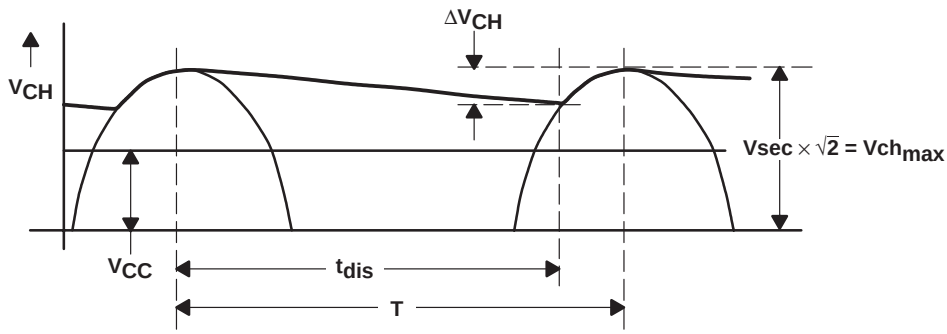


Figure 3–28. Voltages and Timing for the Half-Wave Rectification

- ❏ Advantages
  - Simplified hardware
  - Rectification with the voltage drop of only one diode
- ❏ Disadvantages
  - Charge capacitor, C_{CH}, must have doubled capacity compared to full-wave rectification
  - Higher ripple on the dc supply voltage
  - DC flows through the transformer's secondary winding

Figure 3–29 shows the most simple ac driven power supply. The positive half-wave of the transformer's secondary side charges the load capacitor, C_{CH},.

The capacitor's voltage is stabilized with a Zener diode having a Zener voltage equal to the necessary supply voltage V_{CC} of the MSP430.

Two conditions must be met before a final calculation is possible:

$$V_{SECmin} \times \sqrt{2} - \Delta V_{ch} > V_Z$$

and:

$$\frac{T}{2 \times C_{chmin}} < R_V < \frac{V_{SECmin} \times \sqrt{2} - \Delta V_{ch} - V_Z}{I_{AMmax}}$$

The charge capacitor, C_{CH}, must have a minimum capacity:

$$C_{chmin} \geq \frac{T}{2} \times \left(\frac{1}{R_V} + \frac{I_{AM}}{V_{SECmin} \times \sqrt{2} - V_z} \right)$$

The peak-to-peak ripple voltage $V_{N(PP)}$, of the supply voltage, V_{CC}, is:

$$V_{npp} \approx \frac{\frac{V_{CC}}{I_{AM}} \parallel R_z}{R_v + \frac{V_{CC}}{I_{AM}} \parallel R_z}$$

The final necessary secondary voltage, V_{SEC}, of the ac transformer is ($\Delta V_{CH} = 0.1 \times V_{CHmax}$):

$$V_{SECmin} \geq \frac{1}{\sqrt{2}} \times \left(\frac{0.45 \times T \times I_{AM}}{C_{chmin} - \frac{0.45 \times T}{R_v}} + V_z \right)$$

Where:

I_{AM}	Medium system current (MSP430 and peripherals)	[A]
T	Period of the ac frequency	[s]
ΔV_{ch}	Discharge of Cch during time tdis	[V]
V_{CC}	Supply voltage of the MSP430 system	[V]
V_z	Voltage of the Zener diode	[V]
R_z	Differential resistance of the Zener diode	[ΔV/ΔA]
R_v	Resistance of the series resistor	[Ω]
V_{sec}	Secondary (effective) voltage of the transformer (full load conditions)	[V]

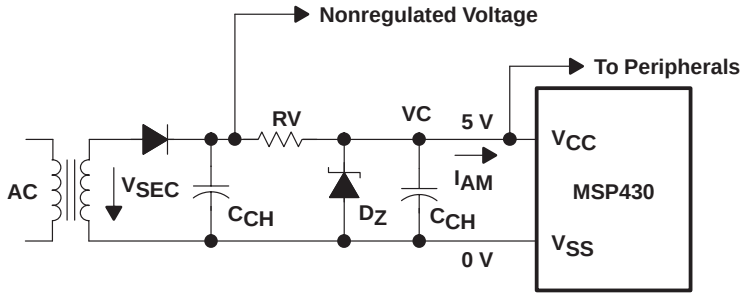


Figure 3–29. Half-Wave Rectification With 1 Voltage and a Zener Diode

Figure 3–30 shows a simplified power supply that uses a voltage regulator like the μA78L05. The charge capacitor, Cch, must have a minimum capacity:

$$C_{chmin} \geq \frac{I_{AM} \times t_{dis}}{\Delta V_{ch}}$$

The peak-to-peak ripple VN(PP) on the output voltage Vreg depends on the used voltage regulator. The regulators ripple rejection value can be seen in its specification. The necessary secondary voltage Vsec of the ac transformer under full load conditions is:

$$V_{SECmin} \geq \frac{1}{\sqrt{2}} \times \left(V_{reg} + V_r + V_d + t_{dis} \times \frac{I_{AM}}{C_{chmin}} \right)$$

The discharge time tdis used with the previous equations is:

$$t_{dis} = T \times \left[1 - \frac{\arccos\left(1 - \frac{\Delta V_{ch}}{V_{SEC} \times \sqrt{2}} \right)}{2\pi} \right]$$

Where:

t_{dis}	Discharge time of Cch	[s]
V_d	Voltage drop of one rectifier diode	[V]
V_r	Dropout voltage (voltage difference between output and input) of the voltage regulator for function	[V]
V_{reg}	Nominal output voltage of the voltage regulator	[V]

For first estimations the value of tdis is calculated for two different discharge values:

- 10% discharge of Cch during tdis tdis = 0.93T
- 30% discharge of Cch during tdis tdis = 0.88T

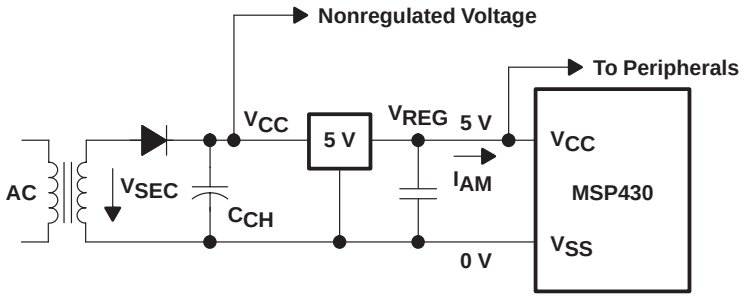


Figure 3–30. Half-Wave Rectification With One Voltage and a Voltage Regulator

Figure 3–31 shows an MSP430 system that uses two supply voltages: +5 V and –5 V. The negative supply voltage is used for analog interfaces. Simple resistor dividers interface the 10-V analog part into the 5 V range of the MSP430. The formulas for the calculation of the charge capacitor, Cch, and the necessary secondary voltage, Vsec, are the same as shown for the circuitry in Figure 3–30. The same circuitry can be used for a system with +2.5 V and –2.5 V (see Figure 3–37 for more details).

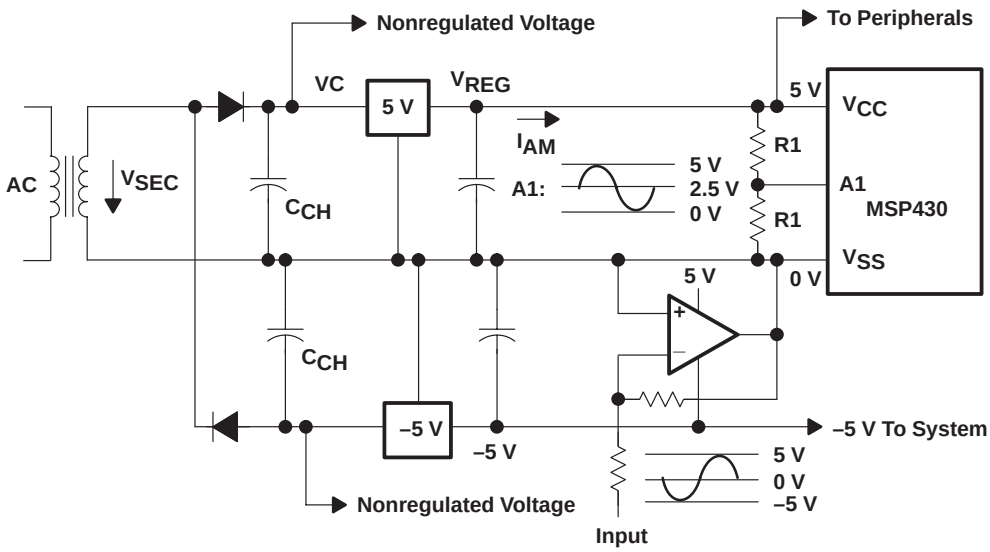


Figure 3–31. Half-Wave Rectification With Two Voltages and Two Voltage Regulators

Full-Wave Rectification

Full-wave rectification uses both half-waves of the secondary voltage, Vsec, for the powering of the application.

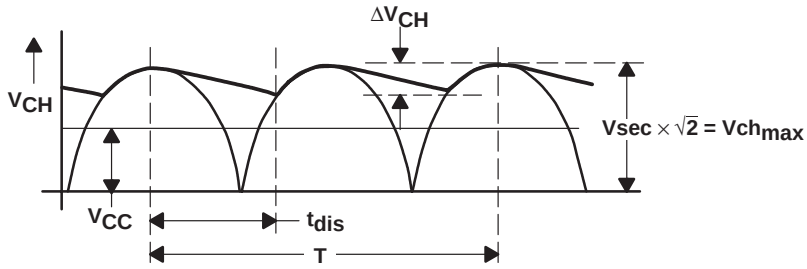


Figure 3–32. Voltages and Timing for Full-Wave Rectification

- Advantages
  - Smaller charge capacitor Cch
  - Lower ripple voltage
  - No dc current through transformer's secondary winding
- Disadvantages
  - Four diodes or a transformer with center tap is necessary
  - Voltage drop of two diodes in series (except with a transformer having a center tap)

Figure 3–33 shows a simple power supply that uses a µA78L05 voltage regulator. The charge capacitor, Cch, must have a minimum capacity:

$$C_{chmin} \geq \frac{I_{AM} \times t_{dis}}{\Delta V_{ch}}$$

The peak-to-peak ripple, $Vn_{pp,}$ on the voltage, V_{CC}, depends on the voltage regulator used. The ripple rejection value can be seen in the voltage regulator specification. The necessary secondary voltage, Vsec, of the ac transformer is for the upper rectifier with four diodes (full load conditions):

$$V_{SECmin} \geq \frac{1}{\sqrt{2}} \times \left(V_{reg} + V_r + 2 \times V_d + t_{dis} \times \frac{I_{AM}}{C_{chmin}} \right)$$

For the center tap transformer, Vd, in the previous equation is multiplied by one (1 × Vd). The discharge time tdis used with the previous equations is:

$$t_{dis} = T \times \left[0.5 - \frac{\arccos\left(1 - \frac{\Delta V_{ch}}{V_{SEC} \times \sqrt{2}} \right)}{2\pi} \right]$$

For first estimations the value of tdis is calculated for two different discharge values:

- ❑ 10% discharge of Cch during tdis tdis = 0.43T
- ❑ 30% discharge of Cch during tdis tdis = 0.38T

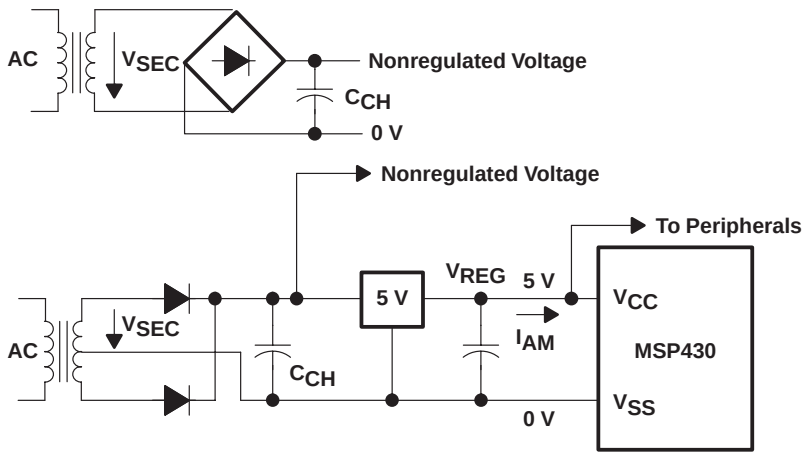


Figure 3–33. Full Wave Rectification for one Voltage with a Voltage Regulator

Figure 3–34 shows an MSP430 system that uses two supply voltages: +2.5 V and –2.5 V. The formulas for the calculation of the charge capacitor, Cch, and the necessary secondary voltage, Vsec, are the same as given for the circuitry in Figure 3–33. The circuitry of Figure 3–34 can also be used for a system with +5-V and –5-V supply (see Figure 3–31 for more details).

Also shown, is how to connect a TRIAC used for ac motor control. The relatively high gate current needed is taken from the non-regulated positive voltage. This reduces the noise within the regulated MSP430 supply. The current flowing through the motor is measured with the ADC for control purposes. The ADC result for 0 V (measured at A0) is subtracted from the current ADC value and results in a signed, offset-corrected value. If a single supply voltage is used (+5V only), the current source can be used to shift the signed current information into the range of the ADC. See Figure 3–27 for the current measurement circuit.

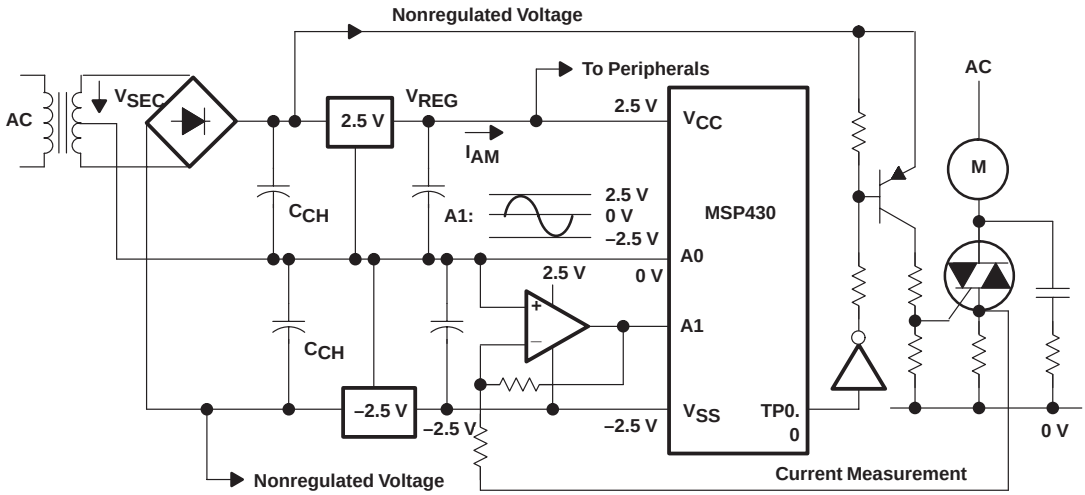


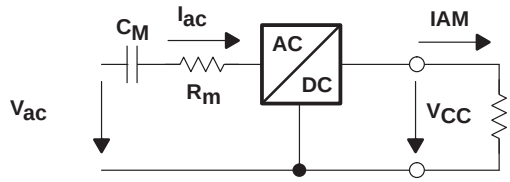
Figure 3–34. Full-Wave Rectification for Two Voltages With Voltage Regulators

3.8.3.2 Capacitor Power Supplies

Applications that do not need isolation from the ac supply or that have a defined connection to the ac supply (like electricity meters) can use capacitor power supplies. The transformer is not needed and only the series capacitor, Cm, must have a high voltage rating due to the voltage spikes possible on ac source.

The ac resistance of the series capacitor, Cm, is used in a voltage divider. This means relatively low power losses. The active power losses are restricted to the protection resistor, Rm, connected in series with Cm. This protection resistor is necessary to limit the current spikes due to voltage spikes and high frequency parts overlaid to the ac voltage. The current I_{ac} through the circuitry is:

$$I_{ac} = \frac{V_{ac}}{\frac{1}{j\omega \times C_m} + R_m} \approx \frac{V_{ac}}{\sqrt{\frac{1}{\omega^2 \times C_m^2} + R_m^2}}$$



Where:

V_{ac}	ac voltage	[V]
f_{ac}	Nominal frequency of the ac	[Hz]
ω	Circle frequency of the ac: $\omega = 2\pi f$	[1/s]
C_m	Series capacitor	[F]
R_m	Series resistor	[Ω]

The previous formula for I_{ac} is valid for all shown capacitor power supplies. The formula assumes low voltages will be generated (< 5% of the ac voltage). For a dc current, I_{AM}, the necessary ac current Iac is:

$$I_{ac} \geq I_{AM} \times \frac{\pi}{\sqrt{2}} = I_{AM} \times 2.221$$

The capacitor, Cm, is:

$$C_{mmin} \geq \frac{1}{2\pi \times f_{acmin}} \times \frac{1}{\sqrt{\left(\frac{V_{acmin} \times \sqrt{2}}{\pi \times I_{AM}}\right)^2 - R_{mmax}^2}}$$

This formula for Cm is valid for all shown capacitor supplies. The calculated value for Cm includes the tolerances for the ac voltage and the ac frequency; the minimum values used for Vac and fac ensure this.

The protection resistor, Rm, for a maximum spike current Imax generated by a voltage spike Vspike is:

$$R_m \geq \frac{V_{spike}}{I_{max}}$$

The charge capacitor, Cch, must have a minimum capacity:

$$C_{chmin} \geq \frac{I_{AM} \times T}{2 \times \Delta V_{ch}}$$

- Advantages
  - No transformer necessary
  - Very simple hardware
- Disadvantages
  - No isolation from ac

Capacitor Supplies for a Single Voltage

Figure 3–35 shows the simplest capacitor power supply. The Zener diode used for limiting the voltage of the charge capacitor, Cch, is used for the voltage regulation too. The peak-to-peak ripple voltage, Vn_{pp}, on voltage, V_{CC}, is:

$$V_{npp} \approx I_{AM} \times \frac{T}{C_{ch} \times 2}$$

The voltage of the Zener diode, Dz, is:

$$V_z \approx V_{CC} + V_d$$

Cch is calculated as shown in Section 3.8.3.2, *Capacitor Power Supplies*.

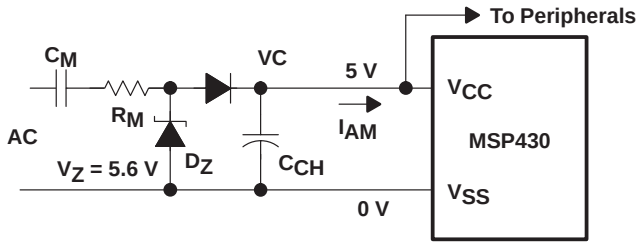


Figure 3–35. Simple Capacitor Power Supply for a Single Voltage

Figure 3–36 shows a hardware proposal for a regulated output voltage, V_{CC}. The voltage, Vz, of the Zener diode, Dz, must be:

$$V_z \geq V_d + V_{reg} + V_r + T \times \frac{I_{AM}}{2 \times C_{chmin}}$$

Cch is calculated as shown in Section 3.8.3.2, *Capacitor Power Supplies*.

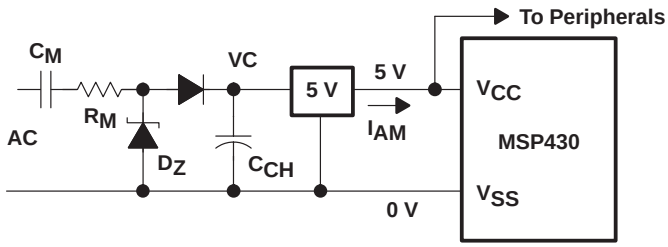


Figure 3–36. Capacitor Power Supply for a Single Voltage

Capacitor Supplies for Two Voltages

Applications that need two voltages (e.g., +2.5 V and –2.5 V) can also use a capacitor supply.

Figure 3–37 shows a split power supply with two regulated output voltages. Together, they deliver the supply voltage, V_{CC}. The split power supply allows the measurement of the voltage of the 0-V line at A0. This value can be subtracted from all other measured analog inputs. This results in offset corrected, signed values. The voltage, Vz, of each Zener diode, Dz, must be:

$$V_z \geq V_r + V_{reg} + T \times \frac{I_{AM}}{2 \times C_{chmin}}$$

The two charge capacitors, Cch, must have the values:

$$C_{chmin} \geq \frac{I_{AM} \times T}{\Delta V_{ch} \times 2}$$

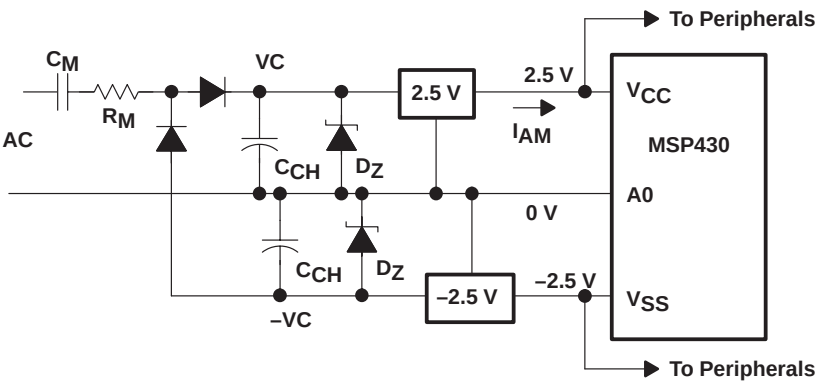


Figure 3–37. Split Capacitor Power Supply for Two Voltages

Figure 3–38 shows a split power supply for +2.5 V and –2.5 V made in a completely different way. It is capable of delivering relatively large output currents due to the buffer transistors. If the high current capability is not needed, the transistors can be omitted and the loads connected to the outputs of the two operational amplifiers directly. The reference for all voltages is a reference diode, LMx85. The highly stable 1.25 V output of this diode is multiplied by two (for +2.5V) or multiplied by –3 and added to the reference value, which delivers –2.5 V. The voltage drop of each one of the two diodes, D, is compensated by the series connection of the two Zener diodes, Dz. The required Zener voltage, Vz, of the two diodes Dz is:

$$V_z \geq \frac{V_{CC}}{2} + V_{BE} + \left(V_{CC} - V_{om}\right) + T \times \frac{I_{AM}}{2 \times C_{chmin}}$$

Where:

V_{BE}	Basis-Emitter voltage of a transistor	[V]
V_{om}	Maximum peak output voltage swing of the operational amplifier with VC	[V]

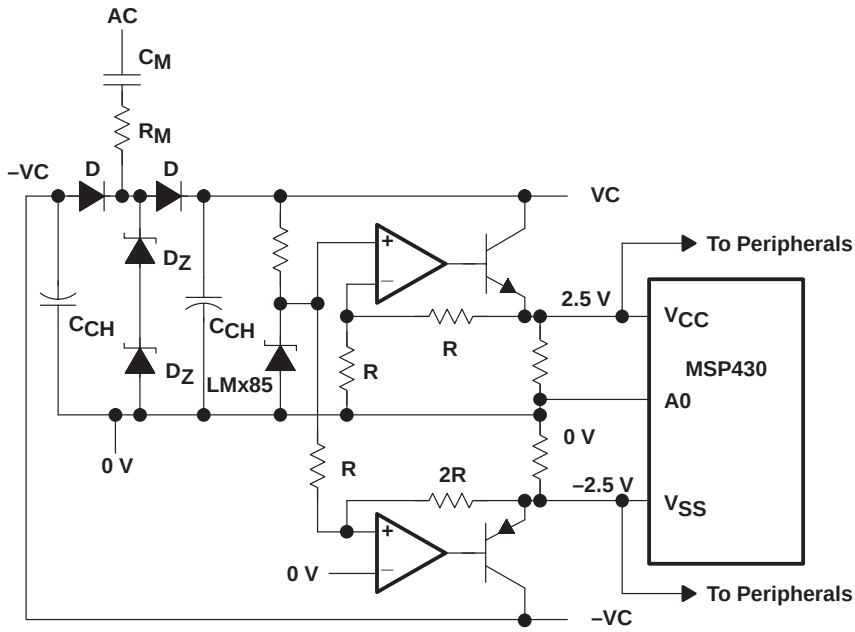


Figure 3–38. Split Capacitor Power Supply for Two Voltages With Discrete Components

3.8.4 Supply From Other System DC Voltages

Existing dc voltages of the controlled system, like +12 V or +24 V, can be used for the supply to the MSP430 system. This is possible due to the low current consumption of the MSP430. So there is nearly no power wasted in the voltage regulator for V_{CC}. If relays and other power consuming peripherals need to be used, the system dc voltage, Vsys, can be used (see Figure 3–40). This solution has two advantages:

- The switching noise is generated outside of the MSP430 supply
- The power for the switched parts does not increase the power of the MSP430 supply

Figure 3–39 and 3–40 show four different possible supplies for an MSP430 system from an existing +12 V (or +24 V) power supply.

3.8.4.1 Zener Diode

A simple configuration of a series resistor R_V with a Zener diode D_Z delivers an output voltage of +3 V or +5 V. The resistor (Rv) is:

$$R_{vmax} > \frac{V_{sysmin} - V_z}{I_{zmin} + I_{AM}}$$

Where:

V_z	Zener voltage of the Zener diode	[V]
I_z	Current through the Zener diode	[A]
V_{sys}	Nominal system voltage	[V]

3.8.4.2 Zener Diode and Operational Amplifier

If larger currents or a higher degree of decoupling is necessary, then an operational amplifier can be used additionally. This way the series resistor Rv can have a much higher resistance than without the operational amplifier. The NPN buffer transistor is only necessary if the operational amplifier cannot output the needed system current. The series resistor Rv is calculated with:

$$R_{vmax} > \frac{V_{sysmin} - V_z}{I_{zmin}}$$

3.8.4.3 Reference Diode With Operational Amplifier

The low voltage of a reference diode (e.g., LMx85) is amplified with an operational amplifier and also buffered. The series resistor, Rv, feeds only the reference diode and has a relatively high resistance. Therefore, it is calculated the same way as shown in Section 3.8.4.2, *Zener Diode*. The output voltage, Vout, is calculated with:

$$V_{out} = V_Z \times \frac{R1 + R2}{R2}$$

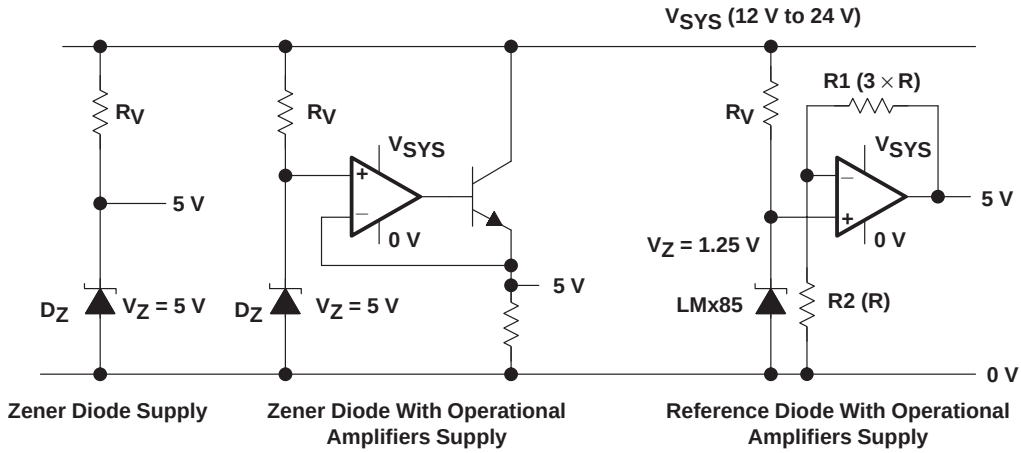


Figure 3–39. Simple Power Supply From Other DC Voltages

3.8.4.4 Integrated Voltage Regulator

Figure 3–40 illustrates the use of an integrated voltage regulator. Here a TPS7350 (regulator plus voltage supervisor) is used, so a highly-reliable system initialization is possible. The TPS7350 also allows the use of the RST/NMI terminal of the MSP430 as described in Section 5.7, *Battery Check and Power Fail Detection*. The RST/NMI terminal is used while running a normal program as an NMI (Non-Maskable Interrupt). This makes possible the saving of important data in an external EEPROM in case of power failure. This is because PG

outputs a negative signal starting at V_{CC} = 4.75 V, which allows a lot of activities until V_{CCmin} of the MSP430 (2.5 V) is reached.

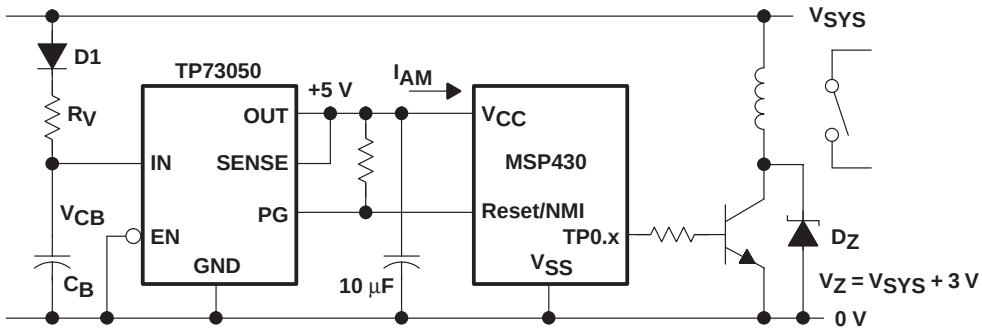


Figure 3–40. Power Supply From Other DC Voltages With a Voltage Regulator

With two additional components (an RC combination) the MSP430 system can be protected against spikes and bridging of supply voltage dropouts is possible. The diode, D1, protects the capacitor, Cb, against discharge during dropouts in voltage Vsys. The series resistor, Rv, is:

$$R_{vmax} < \frac{\left(V_{sysmin} - V_d - V_{CC} - V_r\right)}{I_{AM} + I_{regmax}}$$

The minimum capacity of Cb is:

$$C_{bmin} > \frac{\Delta t \times \left(I_{AM} + I_{reg}\right)}{V_{sysmin} - \left(I_{AM} + I_{regmax}\right) \times R_{vmax} - V_{rmax} - V_{CCmin}}$$

Where:

I_{AM}	System current (medium value MSP430 and peripherals)	[A]
I_{reg}	Supply current of the voltage regulator	[A]
V_{sys}	System voltage (e.g., +12 V)	[V]
V_d	Diode forward voltage (< 0.7 V)	[V]
Vr	Dropout voltage of the voltage regulator for function	[V]
V_{CC}	Supply voltage of the complete MSP430 system	[V]
Δt	Dropout time of V_{sys} to be bridged	[s]

3.8.5 Supply From the M Bus

If the MSP430 system is connected to the M–Bus, three possibilities exist for the supply of the MSP430:

- ❑ Battery Supply: The supply of the MSP430 is completely independent of the M-Bus. This method is not shown in Figure 3–41, because it is the normal way the MSP430 is powered (see Section 3.8.1, *Battery Driven Systems*).
- ❑ M-Bus Supply: The MSP430 system is always supplied by the M-Bus. During off phases of the M-Bus, the MSP430 is not powered.
- ❑ Mixed Supply: Normally the M-Bus supplies the MSP430 and only during off phases of the M-Bus does the battery of the MSP430 provide power.

3.8.5.1 M-Bus Supply

The MSP430 is always powered from the M-Bus. The TSS721 power fail signal, PF, indicates to the MSP430 failure of the bus voltage. This early warning enables the MSP430 to save important data in an external EEPROM. The capacitor, Cch, must have a capacity that allows this storage:

$$C_{chmin} \geq \frac{I_{AM} \times t_{store}}{V_{DD} - V_{CCmin}}$$

Where:

I_{AM}	System current (MSP430 and EEPROM)	[A]
t_{store}	Processing time to store important data into the EEPROM	[s]
V_{DD}	Supply voltage delivered from the TSS721	[V]
V_{CCmin}	Minimum supply voltage of the complete MSP430 system	[V]

3.8.5.2 Mixed Supply

The MSP430 is powered from the M-Bus while bus voltage is available. During times without bus voltage, the battery powers the MSP430. Therefore, a smaller battery can be used when normal bus power is available. The MOS transistor switches to the battery when there is a dropout of the M-Bus voltage. Details are described in the *TSS721 M-Bus Transceiver Application Report*.

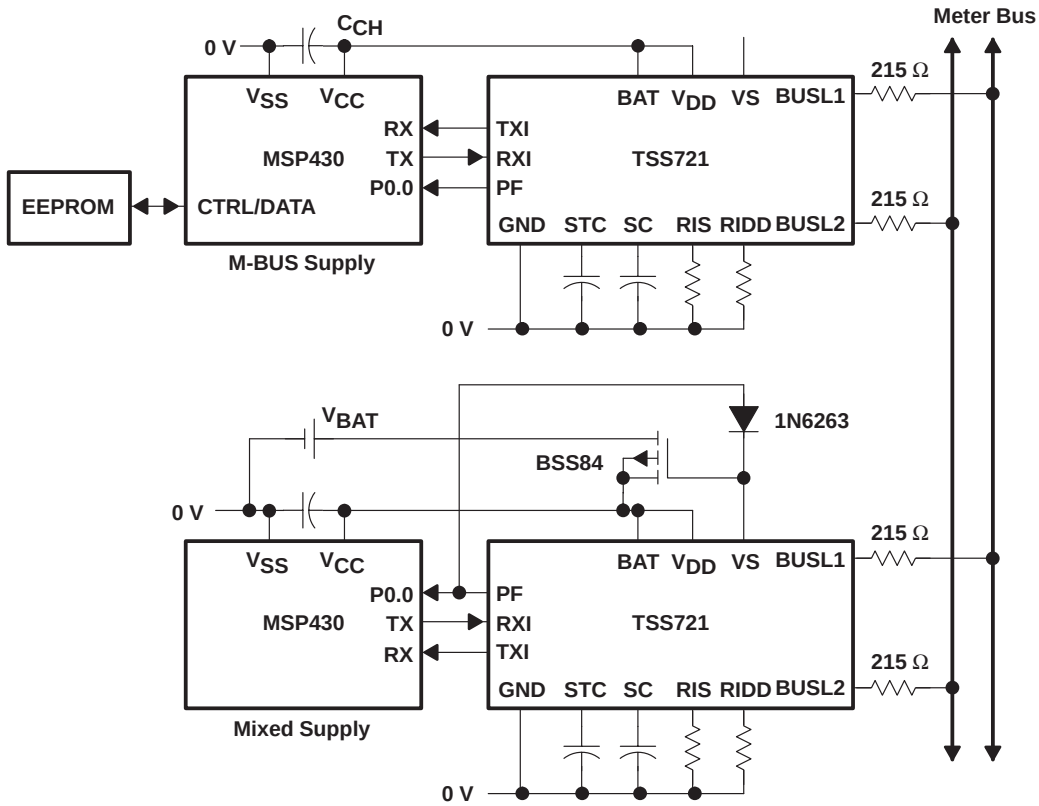


Figure 3–41. Supply From the M Bus

3.8.6 Supply Via a Fiber-Optic Cable

The MSP430 needs a supply current of only 400µA, if supplied with a voltage of 3 V and operating with an MCLK of 1 MHz. This low power can be transmitted via a glass-fiber (fiber-optic) cable. This allows completely isolated measurement systems, which are not possible with other microcomputers. This transmission mode is an advantage for applications in strong electric or magnetic fields.

Because the data transmitted from the host to the MSP430 is also used for the supply of the MSP430 system, a certain amount of light is required continuously and is independent of the transmitted data. Possible ways to reach this are:

- ❑ Use of extended charge periods between host-to-MSP430 data transfers. The MARK level of the RS232 protocol is used for this purpose. This method is shown in Figure 3–42 with every logical one, stop bit, and MARK level used for the supply.
- ❑ Use of a transmission code that always transmits the same number of ones and zeroes, independent of the transmitted data. (e.g., the Bi-Phase Code)

To achieve a positive current balance, a few conditions must be met:

- ❑ The complete hardware design uses ultra-low-power devices (operational amplifiers, reference diode, measurement parts etc.).
- ❑ The MSP430 is in Low Power Mode 3 anytime processing power is not needed.
- ❑ The measurement unit is switched on only during the actual measurement cycle.
- ❑ All applicable hints given in Section 4.9, *Ultra-Low-Power Design with the MSP430 Family* are used.

3.8.6.1 Description of the Hardware

The host sends approximately. 15 mW of optical power into the fiber-optic cable. This optical power is made with a laser diode consuming 30 mW of electrical power. At the other end of the fiber-optic cable, the optical power is converted into 6 mW of electrical power with a power-converter diode. The open-circuit voltage of the power converter (approximately 6 V) decreases to 5 V with the load represented by the MSP430. The received electrical energy is used to charge the capacitor, Cch. The charge-up time required is approximately 300 ms for a capacitor with 30 µF.

The uppermost operational amplifier is used for the voltage regulation of the system supply voltage (3 V to 4 V).

If a stabilized supply voltage is unnecessary, then this operational amplifier can be omitted, as well as the diodes and pull-up resistors at the RST/NMI and P0.1 inputs.

The reference for the complete system is an LMx85 reference diode. This reference voltage (1.25 V) is used for several purposes: trigger threshold for the Schmitt-triggers, reference for the calculation of V_{CC}, and reference for the voltage regulator.

The operational amplifier in the middle works as a reset controller. The Schmitt-trigger switches the RST/NMI input of the MSP430 to a high level when VC reaches approximately 4 V. The RST/NMI input is set low when VC falls below 2.5 V.

The third operational amplifier decodes the information out of the charge voltage and data of the power converter output. This decoder also shows a Schmitt-trigger characteristic.

The measured data is sent back to the host by an IR LED controlled by an NPN transistor. The data format used here is an inverted RS232 protocol and has no current flow for the MARK information (e.g. stop bits).

3.8.6.2 Working Sequence

The normal sequence for a measurement cycle is as follows:

1) The host starts a measurement sequence with the transmission of steady light. This time period is used for the initial charge-up of the charge capacitor, Cch.

2) When this capacitor has enough charge, which means a capacitor voltage (VC) of approximately 4 V is reached, then the reset-Schmitt-trigger switches the RST/NMI input of the MSP430 from low to high. The MSP430 program starts with execution

3) The MSP430 program initializes the system and signals its readiness to the host by the transmission of a defined code via the back channel (a second fiber-optic cable).

4) After the receive of the acknowledge, the host sends the first control instruction (data) to the MSP430.

5) The MSP430 executes the received control instruction and sends back the measured result to the host via the back channel.

6) Items 4 and 5 are repeated as often as needed by the host.

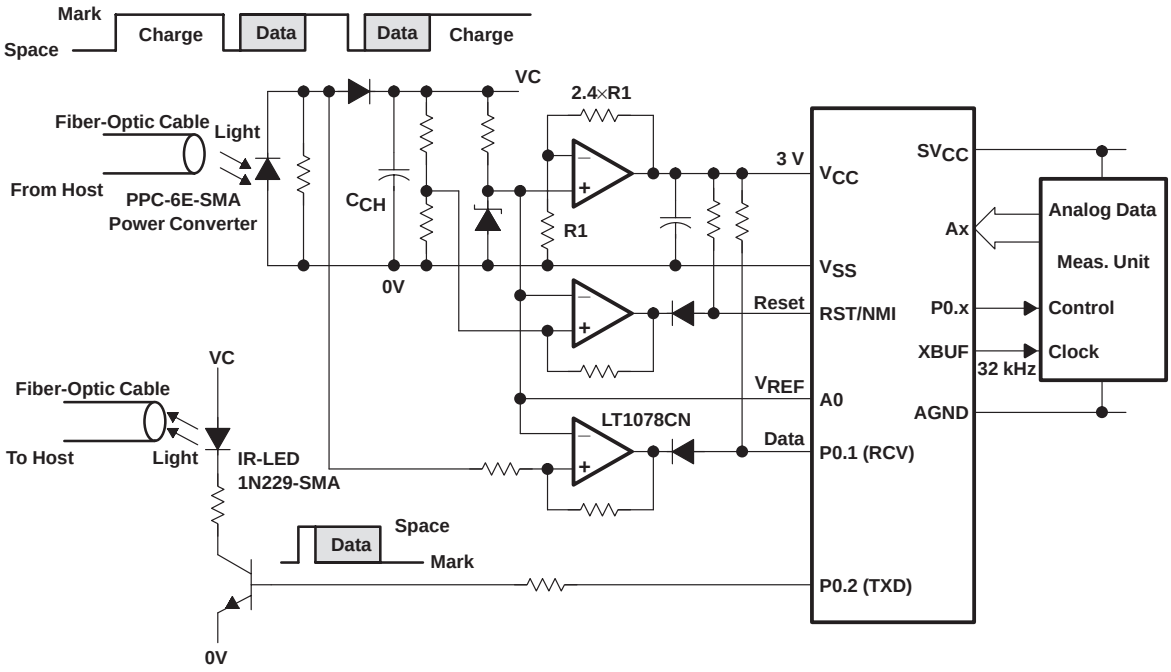


Figure 3–42. Supply via Fiber-Optic Cable

3.8.6.3 Conclusion

The illustrated concepts for supplying the MSP430 family with power demonstrate the numerous ways this can be done. Due to the extreme low power consumption of the MSP430 family, it is possible to supply them with all known power sources. Even fiber-optic cables can be used.