Module 3:

TIMERS AND MIXED SIGNAL SYSTEMS

1. Timers in Embedded Systems

Timers are essential components of embedded systems. They are used for a wide variety of purposes, including generating time delays, controlling time-based events, and measuring the passage of time.

1.1 Watchdog Timer (WDT)

The Watchdog Timer is used to detect and recover from malfunctions in embedded systems. It helps prevent the system from hanging in an infinite loop or deadlock. If the software does not reset the WDT within a specified time period, the system is reset, ensuring it doesn't remain stuck.

- Operation: A timer is set up with a fixed time period. If the software does not reset the timer before it expires, the WDT triggers a reset.
- Example Usage:
    - Periodically reset the watchdog to keep the system running properly.
    - If the system enters an infinite loop, the watchdog will reset the microcontroller.

Example (MSP430):

```
#include <msp430.h>


void watchdog_init() {
   WDTCTL = WDTPW | WDTHOLD;  // Disable the Watchdog Timer
   WDTCTL = WDTPW | WDTCNTCL; // Clear the Watchdog Timer
}


int main() {
   watchdog_init(); // Initialize the watchdog timer


   while(1) {
      // Main loop - reset watchdog timer periodically
      __delay_cycles(100000); // Simulate some work
      WDTCTL = WDTPW | WDTCNTCL; // Reset the Watchdog Timer
```

```
    }

    return 0;
}
```

1.2 Real-Time Clock (RTC)

An RTC is used to keep track of real-time, even during power-down states (using a backup battery). It's critical in systems that need accurate timekeeping, such as data logging or alarms.

- MSP430 RTC Example: The RTC is set to trigger an interrupt at regular intervals to update the system clock or perform other time-dependent tasks.

Example (MSP430):

```
#include <msp430.h>

void RTC_init() {
    RTCCTL = RTCSS_1 | RTCTEV_3;  // Use ACLK and set RTC to trigger every second
    RTCCTL |= RTCIE;  // Enable RTC interrupt
}

#pragma vector=RTC_VECTOR
__interrupt void RTC_ISR(void) {
    // Handle RTC interrupt (e.g., increment time variable)
}

int main() {
    RTC_init();  // Initialize RTC
    __bis_SR_register(GIE);  // Enable global interrupts
    while(1);  // Main loop
    return 0;
}
```

1.3 Timer_A (PWM Generation & Capture Mode)

Timer_A is a flexible timer found in MSP430 and other microcontrollers. It can be used for multiple purposes such as generating PWM signals, measuring time intervals, and capturing external events.

- PWM Generation: Timer_A can generate Pulse Width Modulation (PWM) signals, commonly used for controlling motors, LEDs, and other peripherals.
- Capture Mode: Timer_A can capture the value of a timer counter at the moment a specific event (e.g., rising or falling edge of a signal) occurs.

PWM Generation Example (MSP430):

```
#include <msp430.h>


void Timer_A_PWM_init() {
   P1DIR |= BIT2;  // Set P1.2 as output (PWM pin)
   TA0CCR0 = 1000;  // Set PWM period
   TA0CCTL1 = OUTMOD_7;  // Set PWM output mode
   TA0CCR1 = 500;  // Set 50% duty cycle
   TA0CTL = TASSEL_2 | MC_1;  // Use SMCLK, up mode
}


int main() {
   Timer_A_PWM_init();
   while(1);  // Main loop
   return 0;
}
```

2. Mixed Signal Systems

In embedded systems, mixed signal systems deal with both analog and digital signals. These systems use comparators, analog-to-digital converters (ADC), and other mixed signal peripherals to interface with the real world.

2.1 Comparator_A

A comparator compares two input voltages and outputs a signal based on the comparison result. The comparator can be used to trigger an interrupt when the input voltage crosses a threshold.

- Comparator_A on MSP430 microcontrollers is used to monitor external voltages, for example, to detect if a signal crosses a threshold.

Comparator_A Example (MSP430):

```
#include <msp430.h>
```

```
void Comparator_A_init() {
   CACTL1 = CARSEL | CAF | CAIE;  // Set the threshold, enable interrupt
   CACTL2 = P2 | P1 | P0;  // Configure input pins for the comparator
}

#pragma vector=COMPARATORA_VECTOR
__interrupt void Comparator_A_ISR(void) {
   // Handle the comparator interrupt (e.g., toggle LED)
}

int main() {
   Comparator_A_init();
   __bis_SR_register(GIE);  // Enable global interrupts
   while(1);  // Main loop
   return 0;
}
```

2.2 ADC10 - Architecture and Operation

The ADC10 is a 10-bit analog-to-digital converter commonly used in embedded systems to digitize analog signals. It provides a digital value corresponding to the analog input voltage.

- Single Conversion: The ADC10 performs a single conversion of an analog signal and stores the result in a data register.
- Temperature Sensor: The ADC10 has an internal temperature sensor that can be used to measure the temperature of the microcontroller.

ADC10 Configuration Example (MSP430):

```
#include <msp430.h>

void ADC10_init() {
   ADC10CTL1 = INCH_0;  // Select input channel (e.g., pin A0)
   ADC10CTL0 = SREF_0 | ADC10SHT_3 | ADC10ON | ADC10IE;  // Enable ADC
}
```

```
int main() {
   ADC10_init();  // Initialize ADC10
   while(1) {
      ADC10CTL0 |= ENC | ADC10SC;  // Start conversion
      while (ADC10CTL1 & BUSY);    // Wait for conversion to finish
      unsigned int result = ADC10MEM;  // Read conversion result
   }
   return 0;
}
```

2.3 ADC12 – Comparison with ADC10

The ADC12 is a more advanced version of the ADC10, providing a 12-bit resolution (compared to 10 bits for ADC10). The ADC12 is typically used when higher precision is required.

Key Differences between ADC10 and ADC12:

- Resolution: ADC10 provides 10-bit resolution; ADC12 provides 12-bit resolution.
- Speed: ADC12 typically operates at higher speeds and supports multiple channels.
- Voltage Reference: ADC12 usually offers better reference voltage options, improving accuracy.

3. Summary of Key Concepts

1. Watchdog Timer (WDT): Used for detecting and recovering from system malfunctions. It ensures that the microcontroller resets if the program fails to reset the timer periodically.
2. Real-Time Clock (RTC): Keeps track of time in embedded systems. Useful in applications where the system needs to operate based on time or in applications requiring continuous timekeeping.
3. Timer_A: A flexible timer in MSP430 microcontrollers that can generate PWM signals, measure time intervals, and capture external events.
4. Comparator_A: A component used to compare two input voltages and generate a signal when one input exceeds the other.
5. ADC10 and ADC12: Analog-to-Digital Converters used to convert analog signals into digital data. ADC12 provides higher resolution than ADC10 and is used for more precise applications.