

Structures

```
struct tag-name {  
    type member1;  
    type member2;  
    type member3;  
    .  
    .  
    .  
    type memberN;  
} variable-list;
```

-----Example-----

```
void main(void) {  
    struct mystructtag {  
        float myfloat;  
        int myint;  
    } mystruct1,*mystruct2;  
  
    mystruct2 = (struct mystructtag*)malloc(sizeof(struct mystructtag));  
  
    mystruct1.myfloat = 1.2345;  
    mystruct1.myint = 347;  
  
    mystruct2->myfloat = mystruct1.myfloat/34.2;  
    mystruct2->myint = 412;  
}
```

-----Also can define a new type -----

```
typedef struct {  
    unsigned int addr;  
    int cpr;  
    int rollovers;  
    int flags;  
    unsigned int raw;  
    enc_status_t status;  
    float init_rad_coord;  
    float k, k1;  
} enc_t;
```

```
enc_t test1,test2;
```

```
test1.cpr = 1000;  
test2.addr = 22;
```

-----A function and return a structure -----

```
enc_t myfunc(int val1,float val2);
```

Bit-Fields in C

```
struct tag-name {  
    (unsigned or int) member1:size;  
    (unsigned or int) member2:size;  
    (unsigned or int) member3:size;  
    .  
    .  
    .  
    (unsigned or int) memberN:size;  
} variable-list;
```

-----Example-----

```
void main(void) {  
    struct mybitfieldtag {  
        unsigned fourbitval:4;  
        unsigned eightbitval:8;  
        int      fourbitsigned:4;  
    } mybitfield;  
  
    mybitfield.fourbitval = (a number from 0 to 15);  
    mybitfield.eightbitval = (a number from 0 to 255);  
    mybitfield.fourbitsigned = (a number from -8 to 7);  
  
}
```

Union in C

```
union tag-name {  
    type member1;  
    type member2;  
    type member3;  
    .  
    .  
    .  
    type memberN;  
} variable-list;
```

-----Example-----

```
void main(void) {  
    union myuniontag {  
        unsigned char myc[4];  
        unsigned int myunsignedint;  
        int myint;  
    } myunion;
```

```
    myunion.myc[0] = 0x23;  
    myunion.myc[1] = 0x00;  
    myunion.myc[2] = 0x45;  
    myunion.myc[3] = 0xc3;
```

What does myunion.myunsignedint equal???

What does myunion.myint equal ?????? Is it positive or negative

```
}
```

```

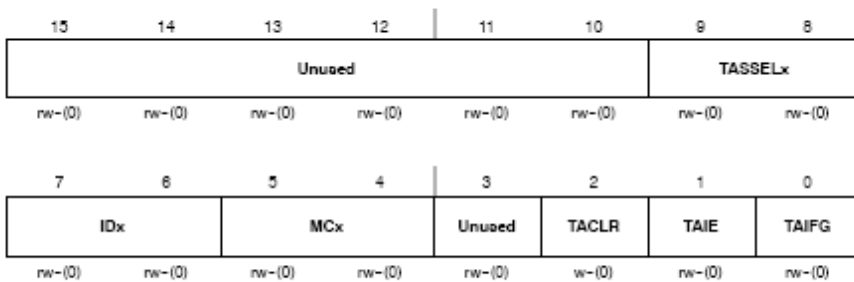
__no_init volatile union
{
  unsigned short TACTL; /* Timer A Control */

  struct
  {
    unsigned short TAIFG      : 1; /* Timer A counter interrupt flag */
    unsigned short TAIE      : 1; /* Timer A counter interrupt enable */
    unsigned short TACLRL    : 1; /* Timer A counter clear */
    unsigned short           : 1;
    unsigned short TAMC      : 2; /* Timer A mode control 0 */
    unsigned short TAID      : 2; /* Timer A clock input divider */
    unsigned short TASSEL    : 2; /* Timer A clock source select */
    unsigned short           : 6;
  } TACTL_bit;
} @ 0x0160;

```

Timer_A Registers

TACTL, Timer_A Control Register



Unused	Bits 15-10	Unused
TASSELx	Bits 9-8	Timer_A clock source select 00 TACLK 01 ACLK 10 SMCLK 11 INCLK
IDx	Bits 7-6	Input divider. These bits select the divider for the input clock. 00 /1 01 /2 10 /4 11 /8
MCx	Bits 5-4	Mode control. Setting MCx = 00h when Timer_A is not in use conserves power. 00 Stop mode: the timer is halted. 01 Up mode: the timer counts up to TACCR0. 10 Continuous mode: the timer counts up to 0FFFFh. 11 Up/down mode: the timer counts up to TACCR0 then down to 0000h.
Unused	Bit 3	Unused
TACLRL	Bit 2	Timer_A clear. Setting this bit resets TAR, the clock divider, and the count direction. The TACLRL bit is automatically reset and is always read as zero.
TAIE	Bit 1	Timer_A interrupt enable. This bit enables the TAIFG interrupt request. 0 Interrupt disabled 1 Interrupt enabled
TAIFG	Bit 0	Timer_A interrupt flag 0 No interrupt pending 1 Interrupt pending

```

#include <stdarg.h>
#include <stdio.h>
// #include <msp430x20x2.h>
#include <in430.h>
#include <io430.h>

void main(void)
{

    WDTCTL = WDTPW + WDTHOLD; // Stop WDT Can't use bit Union because of Password
    DCOCTL = CALDCO_16MHZ; // Set uC to run at approximately 16 Mhz
    BCSCTL1 = CALBC1_16MHZ;

    // P1DIR |= 0x81; // P1.0 and P1.7 outputs
    P1DIR_bit.P1DIR_0 = 1;
    P1DIR_bit.P1DIR_7 = 1;

    // CCTL0 = CCIE; // CCIE = 0x0010 CCR0 interrupt enabled
    TACCTL0_bit.CCIE = 1;

    CCR0 = 1653; // 1653 is 1/9600 seconds sets up timer for output only UART baud rate of 9600.
    // TACTL = TASSEL_2 + MC_2; // TASSEL_2 = 0x200 and MC_2 = 0x20 SMCLK, contmode
    TACTL_bit.TASSEL = 2;
    TACTL_bit.TAMC = 2;

    _BIS_SR(GIE); // Enable global interrupt

    while(1) {
        if (newprint == 1) { // newprint set to one in timer ISR

            // P1OUT ^= 0x01; // toggle P1.0 on and off
            P1OUT_bit.P1OUT_0 ^= 1;

            UART_printf("Hello %d\n\r",timeint/freq); // Print to UART
            newprint = 0; // reset flag back to zero

        }
    }
}

```

Some Structures and Unions that are Used in the Robot's Vision Code

```
union pixeldata {
    int pixelquad;
    struct pixels {
        unsigned char UthenYk;
        unsigned char UthenYkp1;
        unsigned char UthenYkp2;
        unsigned char UthenYkp3;
    } pixels;
};

typedef struct bgr {
    unsigned char blue;
    unsigned char green;
    unsigned char red;
} bgr;

bgr **ptrImage
ptrImage[0][0].red = 128;
ptrImage[45][23].blue = 134;

typedef struct hsvtag {
    unsigned char h;
    unsigned char s;
    unsigned char v;
} hsv;

bgr **ptrHSVImage
ptrHSVImage[3][2].h = 20;
ptrHSVImage[65][12].v = 160;

#define MAX_NUM_OBJECTS 20
struct final_object_stats{
    int sum_r;
    int sum_c;
    float C11_sum;
    float C20_sum;
    float C02_sum;
    int num_pixels_in_object;
    float center_r;
    float center_c;
    float center_x;
    float center_y;
    float theta;

} final_object_stats[MAX_NUM_OBJECTS+1];
```