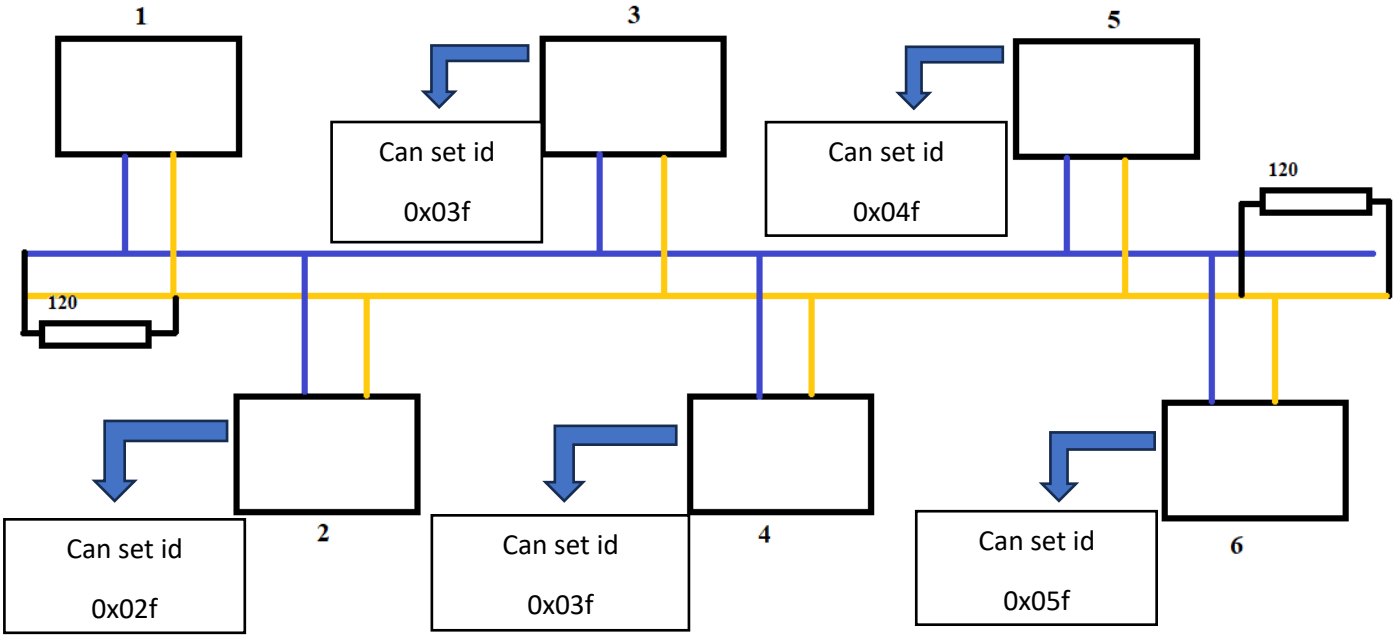
ข้อมูลเพื่อใช้ประกอบการฝึกอบรมพอสังเขป



สรุปหน้าที่การทำงานของแต่ละ NODE

-NODE 1

รับข้อมูล อุณหภูมิมอเตอร์ อุณหภูมิชุดขับเคลื่อนมอเตอร์ จาก NODE 4

รับข้อมูล สัญญานคันเร่ง ความเร็วรอบมอเตอร์ จาก NODE 5

รับข้อมูล แรงดันแบตเตอรี่ ค่ากระแสมอเตอร์ จาก NODE 6

ส่งข้อมูล การควบคุมความเร็วมอเตอร์ให้กับ NODE 5

ส่งข้อมูล การควบคุมทิศทางมอเตอร์ให้กับ NODE 6

NODE 2

NODE 3

NODE 4

ส่งข้อมูล อุณหภูมิมอเตอร์ อุณหภูมิชุดขับเคลื่อนมอเตอร์ ให้ NODE 1

NODE 5

ส่งข้อมูล สันญานคันเร่ง และ ความเร็วมอเตอร์ให้กับ NODE 1

รับข้อมูลโหมดการควบคุมจาก NODE 1

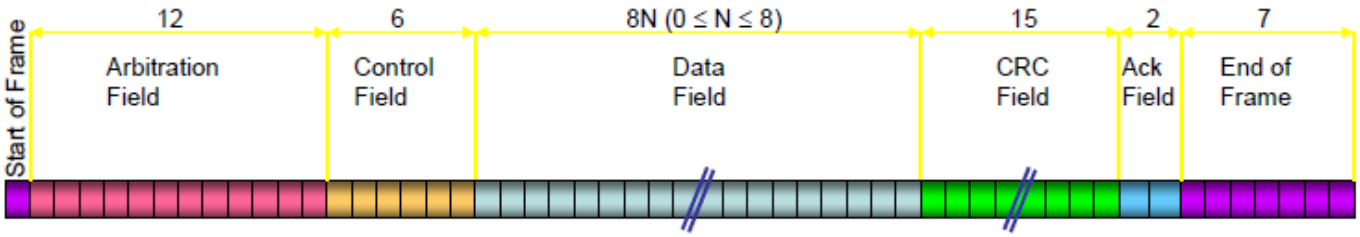
NODE 6

ส่งข้อมูลแรงดันแบตเตอรี่ และ ค่ากระแสมอเตอร์ให้กับ NODE 1

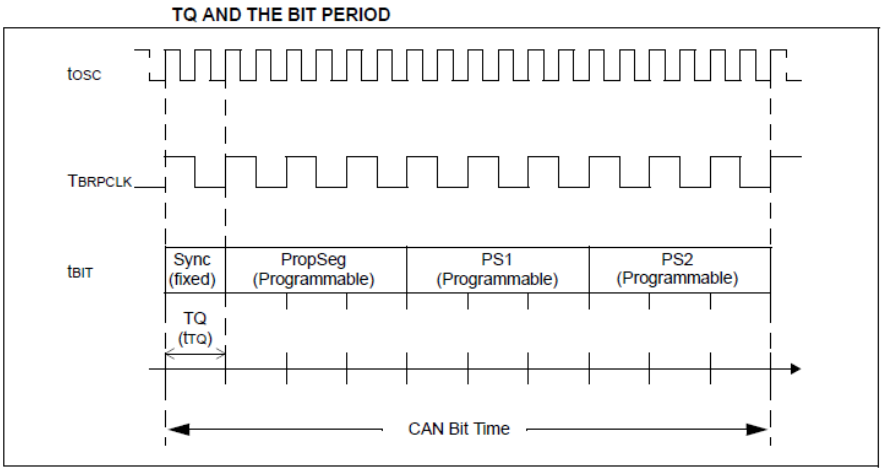
รับข้อมูลการควบคุมทิศทางมอเตอร์จาก NODE 1

ใช้รูปแบบ Standard Data Frame ในการรับส่งข้อมูล

- Versions 1.0 and 2.0A
- 11-bit Identifier Field



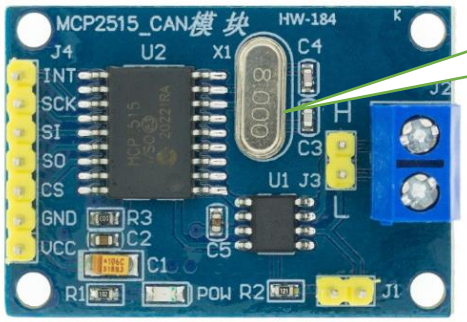
ใช้ความเร็วในการรับส่งข้อมูล(Baue rate) 62.5Kbit/sec



Bit time period

การคำณวนค่าความเร็วในการสื่อสาร

Tq = (2 x (PRE + 1))/Fosc)



Fosc

8Mhz

Tq = (2x(3+1))/8000000

= 1usec

#define CAN_USE_EXTENDED_ID false

#define CAN_BRG_SYNCH_JUMP_WIDTH 0 **//synchronized jump width (def: 1 x Tq)**

#define CAN_BRG_PRESCALAR 3

Prescale=3

#define CAN_BRG_SAM 0

#define CAN_BRG_PHASE_SEGMENT_1 5 **//phase segment 1 (def: 6 x Tq)**

#define CAN_BRG_PROPAGATION_TIME 2 **//propagation time select (def: 3 x Tq)**

#define CAN_BRG_WAKE_FILTER FALSE

```
#define CAN_BRG_PHASE_SEGMENT_2  5 //phase segment 2 time select (def: 6 x Tq)

#define CAN_USE_RX_DOUBLE_BUFFER TRUE

#define CAN_ENABLE_DRIVE_HIGH 0

#define CAN_ENABLE_CAN_CAPTURE 0
```

จาก source code เวลาใน 1 bit = 1+6+3+6 = 16 Tq = 16 usec

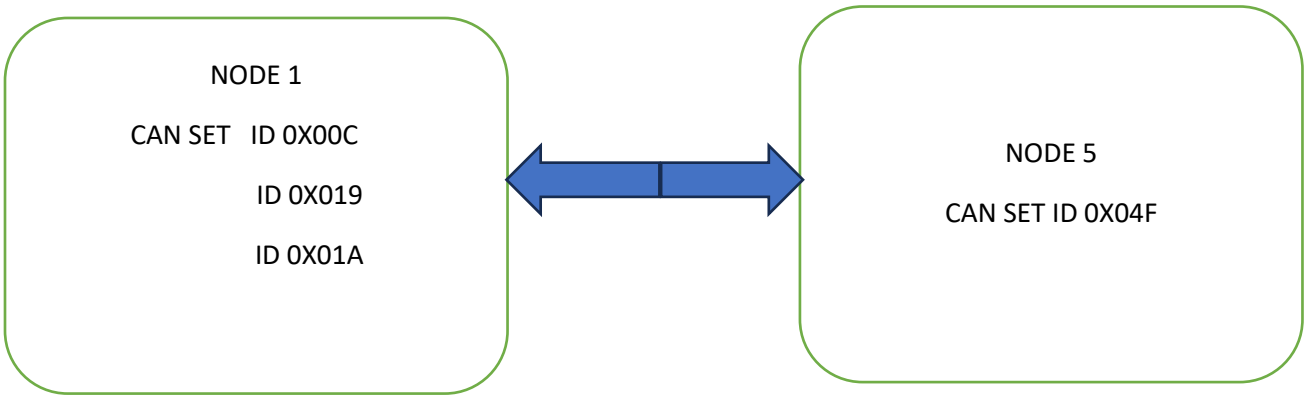
ดังนั้นจะได้ความเร็วในการสื่อสาร(Baue rate)

= 1/16 usec

= 1/0.000016

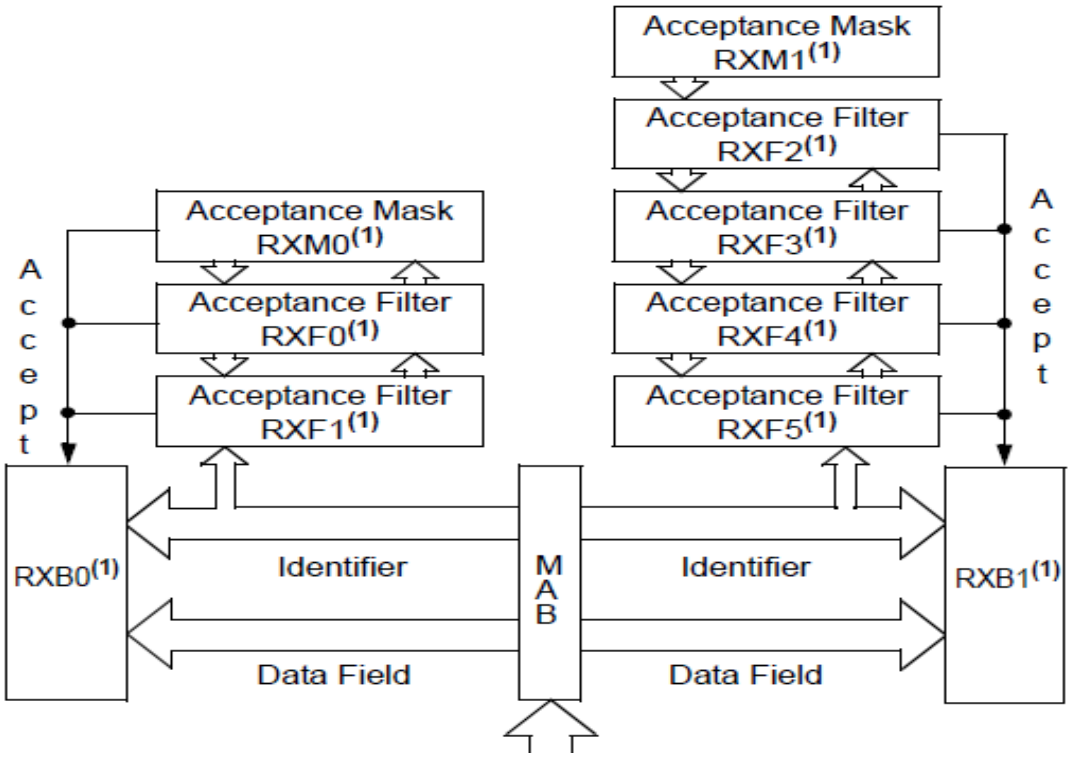
= 62500 bit/sec

ตัวอย่างการรับสังข้อมูลระหว่าง NODE 1 กับ NODE 5



จากรูปข้างบนกล่าวได้ว่ากำหนดให้ NODE 1 รับข้อมูลรับข้อมูลจาก 3 ID คือ 0X00C,0X019 และ 0X01A ซึ่งมาจากกา

การกำหนดค่า หนด ให้กับส่วน ของ register mask และ register filter



Source code การกำหนดค่าให้กับ register mask และ register filter ให้กับ NODE 1 เพื่อให้รับข้อมูลจาก ID

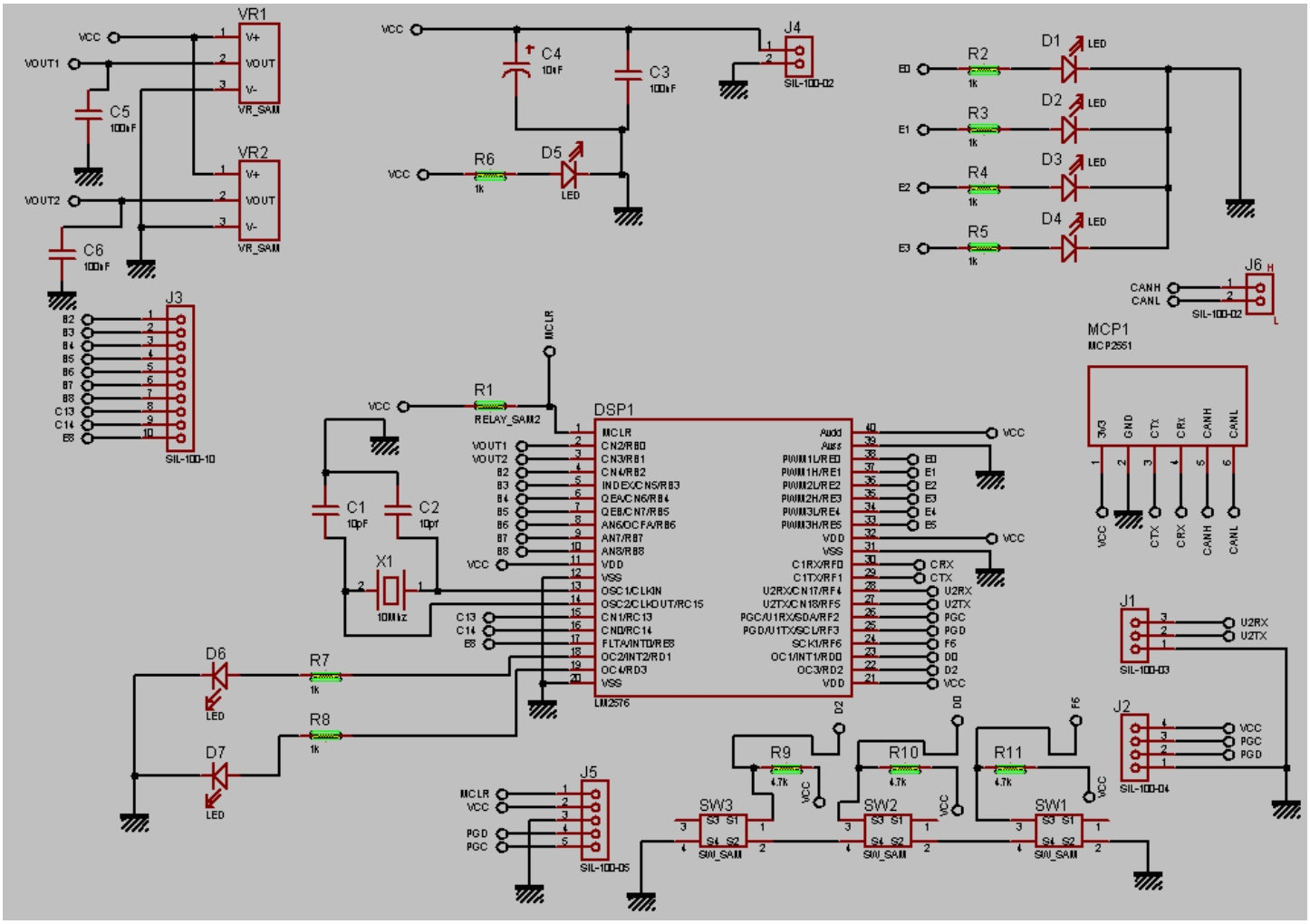
0X00C,0X019 และ 0X01A

```
can_set_id(&C1RXM0, 0B11111111110, 0);  //set mask 0
 can_set_id(&C1RXF0, 0B00000011010, 0);  //set filter 0 of mask 0
 can_set_id(&C1RXF1, 0B00000001100, 0);  //set filter 1 of mask 0
 can_set_id(&C1RXM1, 0B11111111111, 0);  //set mask 1
 can_set_id(&C1RXF2, 0B00000011001, 0);  //set filter 0 of mask 1
 can_set_id(&C1RXF3, 0B00000011001, 0);  //set filter 1 of mask 1
 can_set_id(&C1RXF4, 0B00000011001, 0);  //set filter 2 of mask 1
 can_set_id(&C1RXF5, 0B00000011001, 0);  //set filter 3 of mask 1
```

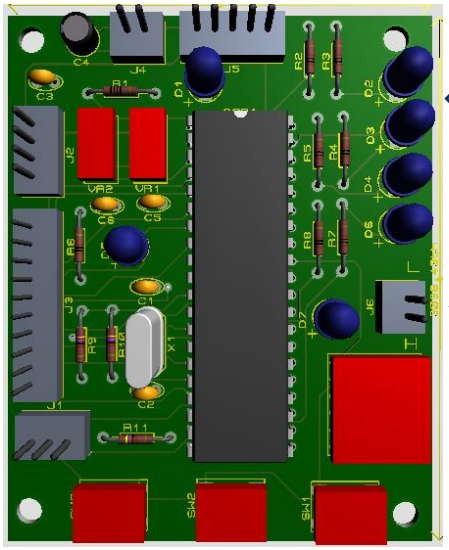
source code การกำหนดค่าให้กับ register mask และ register filter ให้กับ NODE 5 เพื่อให้รับข้อมูลเฉพาะ ID0X04F

```
 can_set_id(RX0MASK, 0xfff, 0);  //set mask 0 (RX BUFFER 0)
 can_set_id(RX0FILTER0, 0x04f, 0);  //set filter 0 of mask 0 (RX BUFFER 0)
 can_set_id(RX0FILTER1, 0x04f, 0);  //set filter 1 of mask 0 (RX BUFFER 0)
 can_set_id(RX1MASK, 0xfff, 0);  //set mask 1 (RX BUFFER 1)
 can_set_id(RX1FILTER2, 0x04f, 0);  //set filter 0 of mask 1 (RX BUFFER 1)
 can_set_id(RX1FILTER3, 0x04f, 0);  //set filter 1 of mask 1 (RX BUFFER 1)
 can_set_id(RX1FILTER4, 0x04f, 0);  //set filter 2 of mask 1 (RX BUFFER 1)
 can_set_id(RX1FILTER5, 0x04f, 0);  //set filter 3 of mask 1 (RX BUFFER 1)
```

รายละเอียดวงจร NODE 1



Touch Screen Display



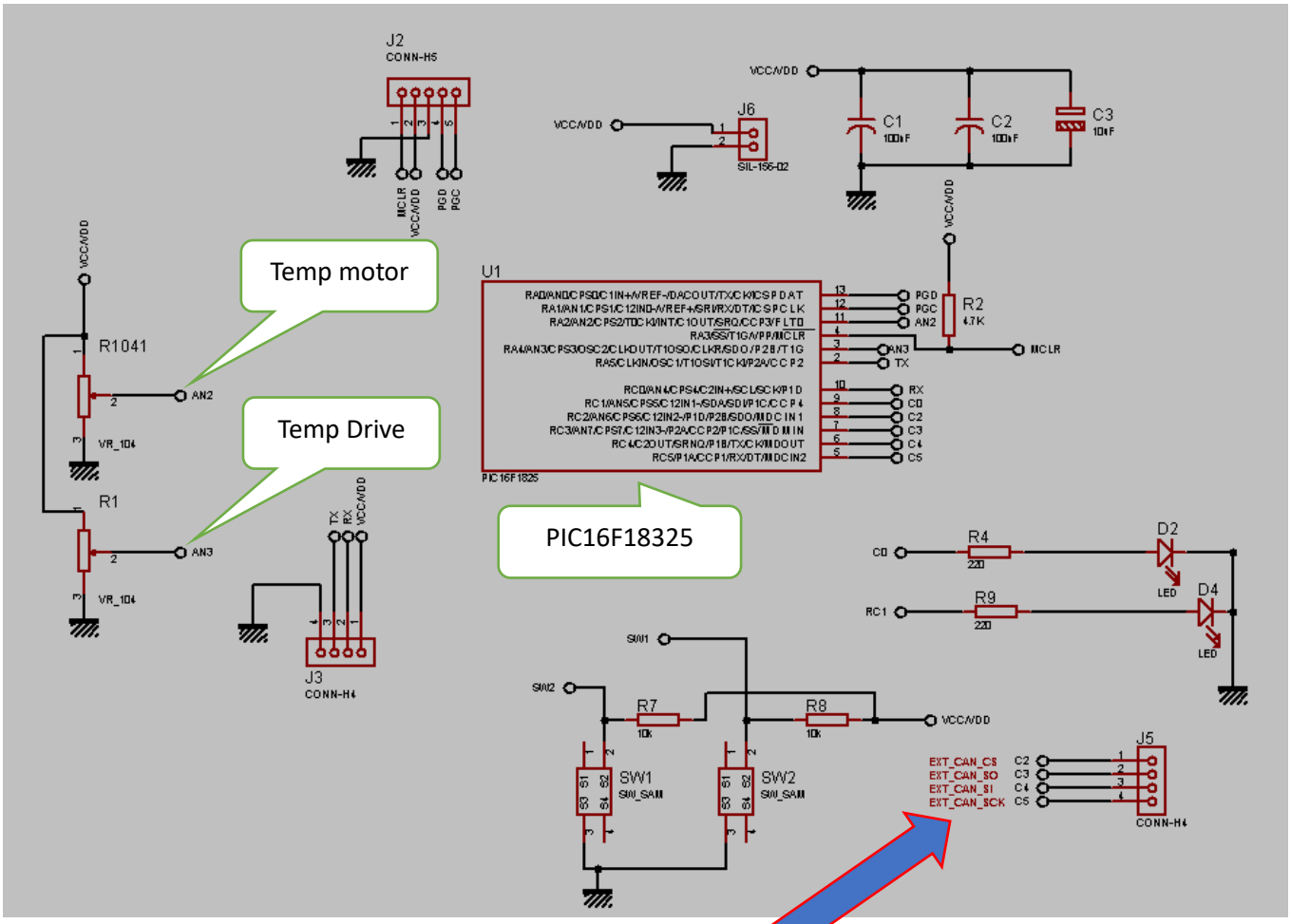
Rs232(2)

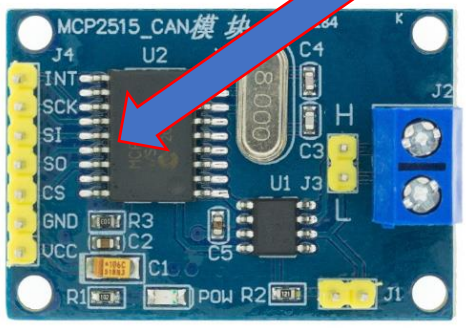
CAN BUS

DSPIC30F4011

NEXTION DISPLAY

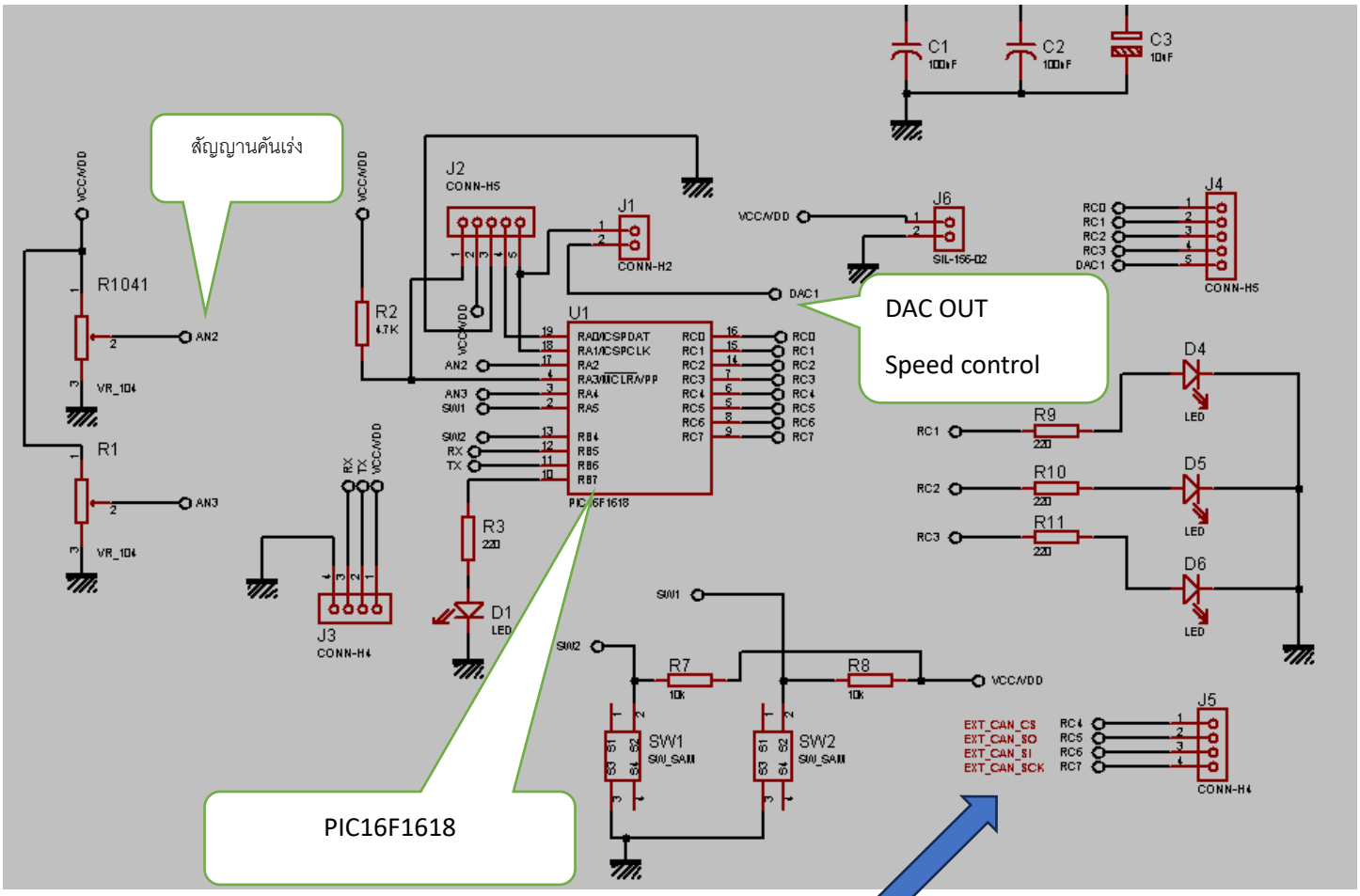
รายละเอียดวงจร NODE 2,3,4,6

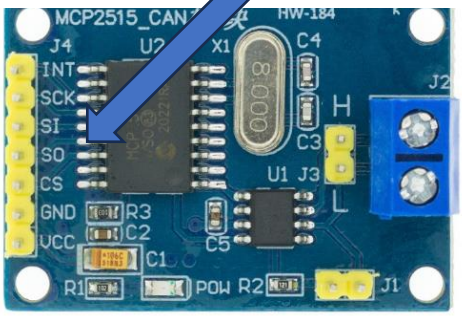




CAN BUS

รายละเอียดวงจร NODE 5





CAN BUS

Source code NODE 1 ประกอบด้วย 4 file คือ

1 Node_c.c

```
#include "30F4011.h"
#device ICSP=1
#device PASS_STRINGS = IN_RAM
#include<string.h>
#include<stdio.h>
#include<stdlib.h>
#priority RDA2
#FUSES HS
#FUSES NOBROWNOUT              //No brownout reset
#use delay(crystal=10000000)
//#use delay(CLOCK=20MHZ)
#use rs232(UART1, baud=9600,PARITY=n,BITS =8,STOP=1, stream=PORT1)
#use rs232(xmit=PIN_f5, rcv=PIN_f4, baud=9600,PARITY=n,BITS =8,STOP=1, stream=PORT2)
//#define CAN_USE_EXTENDED_ID   false
#define CAN_USE_EXTENDED_ID        false
 #define CAN_BRG_SYNCH_JUMP_WIDTH  0
 #define CAN_BRG_PRESCALAR  4
 #define CAN_BRG_SAM 0
 #define CAN_BRG_PHASE_SEGMENT_1  5
 #define CAN_BRG_PROPAGATION_TIME 2
 #define CAN_BRG_WAKE_FILTER FALSE
 #define CAN_BRG_PHASE_SEGMENT_2 5
 #define CAN_USE_RX_DOUBLE_BUFFER TRUE
 #define CAN_ENABLE_DRIVE_HIGH 0
  #define CAN_ENABLE_CAN_CAPTURE 0
   #include "can-dsPIC30.c"
   struct rx_stat rxstat;
   int32 rx_id;
  unsigned int8 in_data[8];
   int8 rx_len;
   int8 out_data[8];
   int32 tx_id=27;
   int1 tx_rtr=0;
   int1 tx_ext=0;
   int8 tx_len=8;
   int8 tx_pri=1;
   char str[16];
   char str2[15];
    char str3[15];
    char ccc[10]={'0','1','2','3','4','5','6','7','8','9'};
   char send_display[16];
   int8 i;
  // int8 out_data[8];
 unsigned  int8 buffer[8];
   char a,b,c;
   int8 num0,num1,num2;
     int8 ms=0;
    unsigned int16 *ptr;
   unsigned int8 speed1=0;
     int8 speed2;
```

```
//////////////////////////////VAR NEXTION ////////////
signed long result;
char *ptrx;
   char data_ser[30];
   signed int8 index_byte=0;
   int1 flag_rx=0;
   int1 flg_disable_rda=0;
   int1 flag_rda=0;
   int1 flag_page_display=0;
   int1 flag_obd=0;
   int1 flag_motor=0;
   int8 k=0;
  VOID SENDSTRING(CHAR *TXT,int8 num_txt);
  VOID SENDSTRING_display(CHAR *TXT,int8 num_txt);
   VOID SENDSTRING_obd(CHAR *TXT,int8 num_txt);
     VOID SENDSTRING_motor(CHAR *TXT,int8 num_txt);
  void process_display();
   void process_obd();
    void process_motor(void);
   #include "instrument.c"
   #include "check.c"
   #include "motor.c"
   void process_command();
    void process_command2();
    void page_display(void);
  #include "monitor.c"
#INT_TIMER2
void  timer2_isr(void)
{
 ms++;//output_toggle(pin_e3);
}
#INT_RDA2
void  rda2_isr(void)
{
//////////////////////////////////////////get data from nextion display//////////////////////

          flag_rda=1;

          data_ser[index_byte] =fgetc(port2);//output_toggle(pin_e8);
           if (( index_byte>3) && ((unsigned char)data_ser[index_byte]==0xff) && ((unsigned char)data_ser[index_byte-
1]==0xff)&&((unsigned char)data_ser[index_byte-2]==0xff) )
          {
           flag_rx=1;//data serial ok
           disable_interrupts(INT_RDA2);//flg_disable_rda=1;
           // fprintf(port1,data_ser);
          // process_command();
          strcpy (str2, data_ser);
          }
     index_byte++;
    if(index_byte>15)
    {
     index_byte=0;enable_interrupts(INT_RDA2);flg_disable_rda=0;

     flag_rx=0;memset(data_ser,0,30);

    }

}
```

```
void main()
{
    can_init();
   // setup_timer2(TMR_INTERNAL | TMR_DIV_BY_1, 5000);//over flow 2msec
     disable_interrupts(INT_TIMER2);
    // enable_interrupts(INT_RDA2);
     enable_interrupts(INTR_GLOBAL);
      DELAY_ms(1000);
     buffer[0] = 10;      buffer[1] = 20;
     buffer[2] = 30;      buffer[3] = 40;
     buffer[4] = 50;      buffer[5] = 60;
     buffer[6] = 70;      buffer[7] = 80;
     i=0;
  for (i=0;i<8;i++) {
   output_low(pin_e2);output_low(pin_e1);output_low(pin_e0);
   delay_ms(500);
   output_high(pin_e2);output_high(pin_e1);output_high(pin_e0);
   delay_ms(500);
  }
  memset(data_ser,0,30);enable_interrupts(INT_RDA2);
output_low(pin_e2);output_low(pin_e8);output_low(pin_e4);
  while(true)
  {
      if(flag_rx)
     {flag_rda=0;
      process_command();// page_display();
      flag_rx=0;index_byte=0; memset(data_ser,0,30); enable_interrupts(INT_RDA2);
     }
      output_toggle(pin_e3);delay_ms(500);


  }

}

 void process_command()

 {

    if ( strstr(data_ser,"get_id15" ))

   {

   SENDSTRING("Instrument page",3);

   delay_ms(3000);

  flag_page_display=1;page_display();

  //delay_ms(500);



   }

   else if(strstr(data_ser,"get_id16" ))

   {

   SENDSTRING("Engin page",3);

    delay_ms(3000);

  flag_motor=1;page_motor();
```

```
  }
  else if(strstr(data_ser,"get_id27" ))
  {
  SENDSTRING("obdpage",3);
   delay_ms(3000);
 flag_obd=1;page_obd();

  }
  else{flag_rx=0;index_byte=0; memset(data_ser,0,30);flag_rda=0; enable_interrupts(INT_RDA2);}

}
```