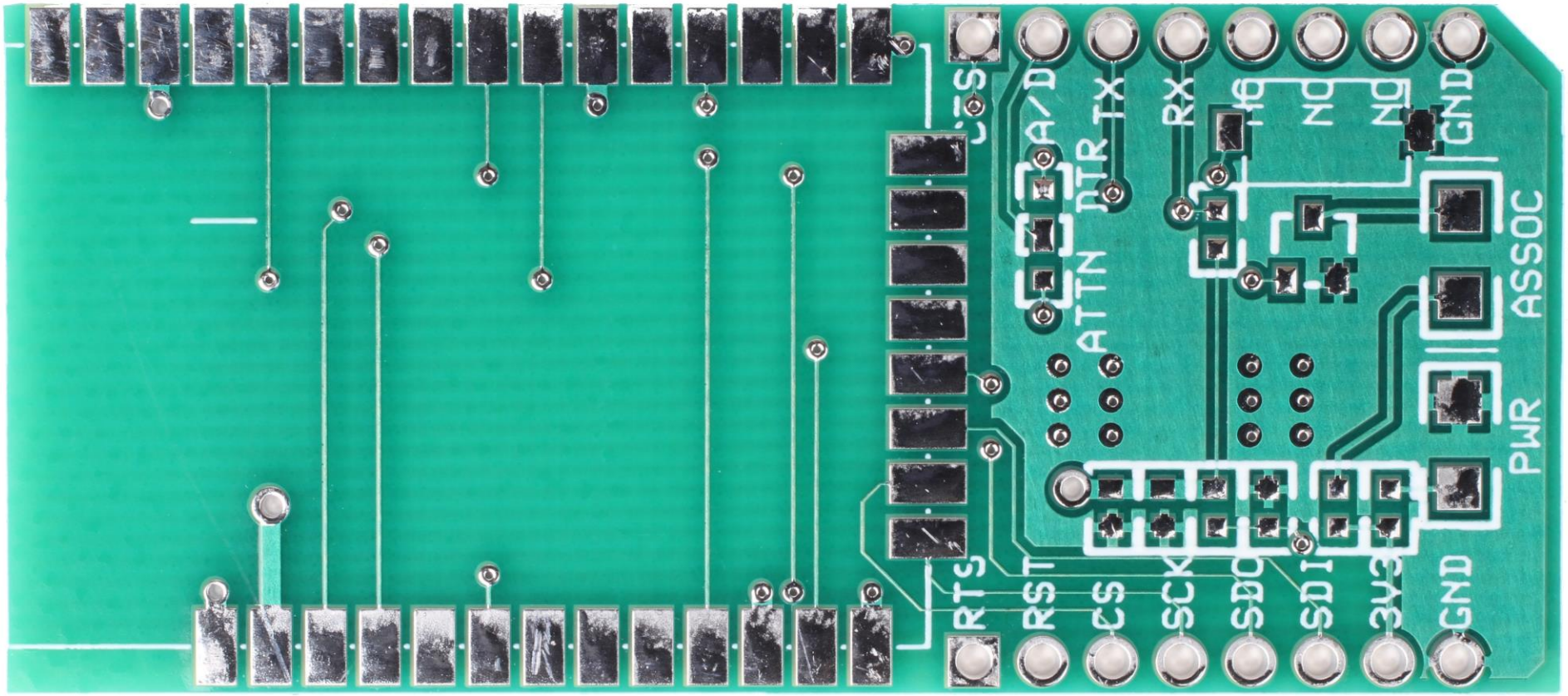
ARM Your Sensors



An ARMED Mobile Sensor Node Reference Design

August 30, 2018

Fred Eady

DesignNews

Presented by:

CEC CONTINUING EDUCATION CENTER

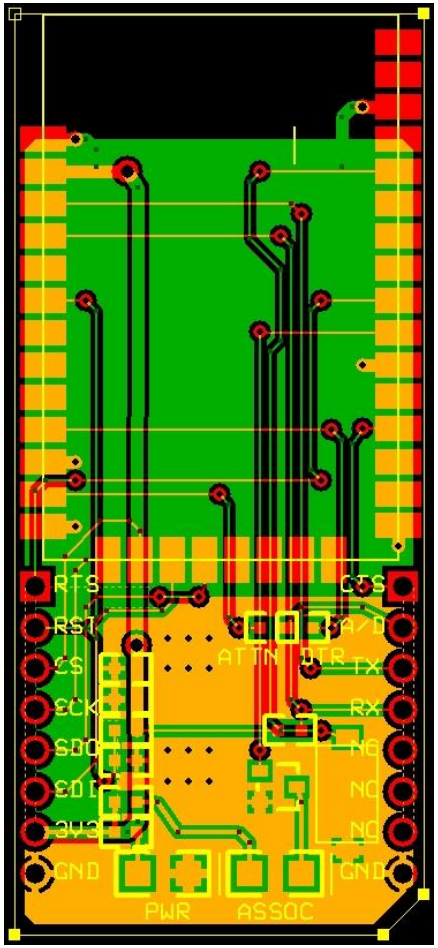
Digi-Key ELECTRONICS

AGENDA

- A Home-Brewed click-compatible XBee Module
- ARMing Our Sensor/XBee click-compatible Module
- Under Pressure
- Day 4 Summary

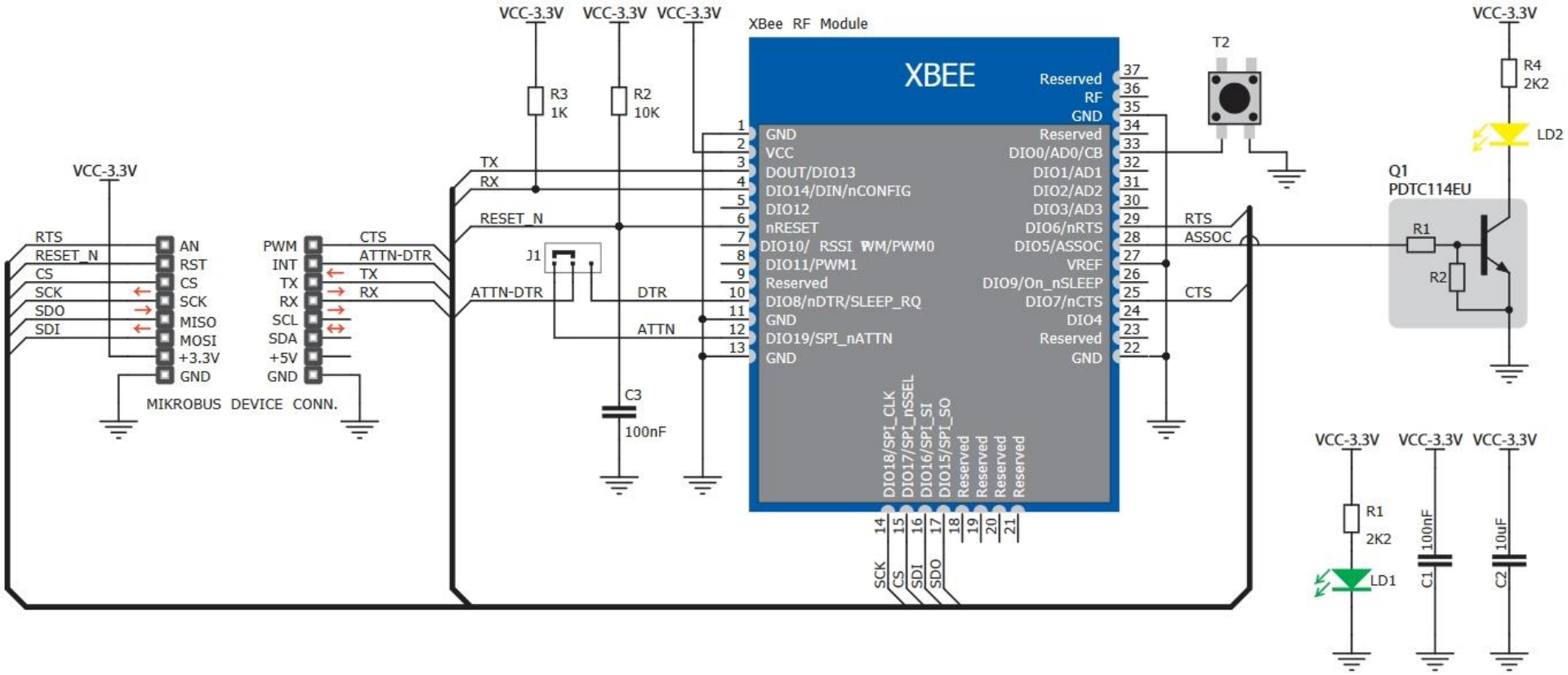
ARM Your Sensors

A Home-Brewed click-compatible XBee Module



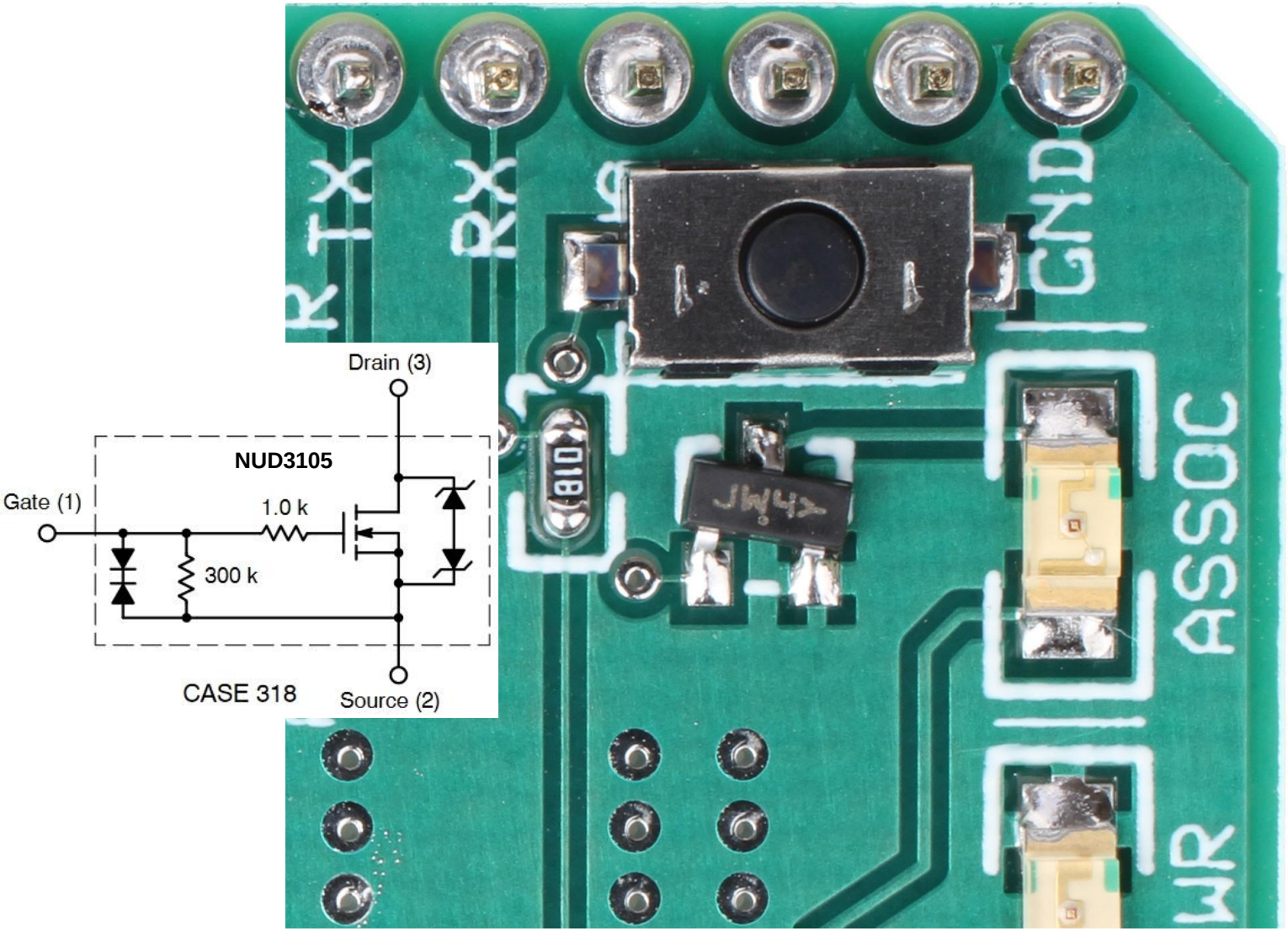
ARM Your Sensors

A Home-Brewed click-compatible XBee Module



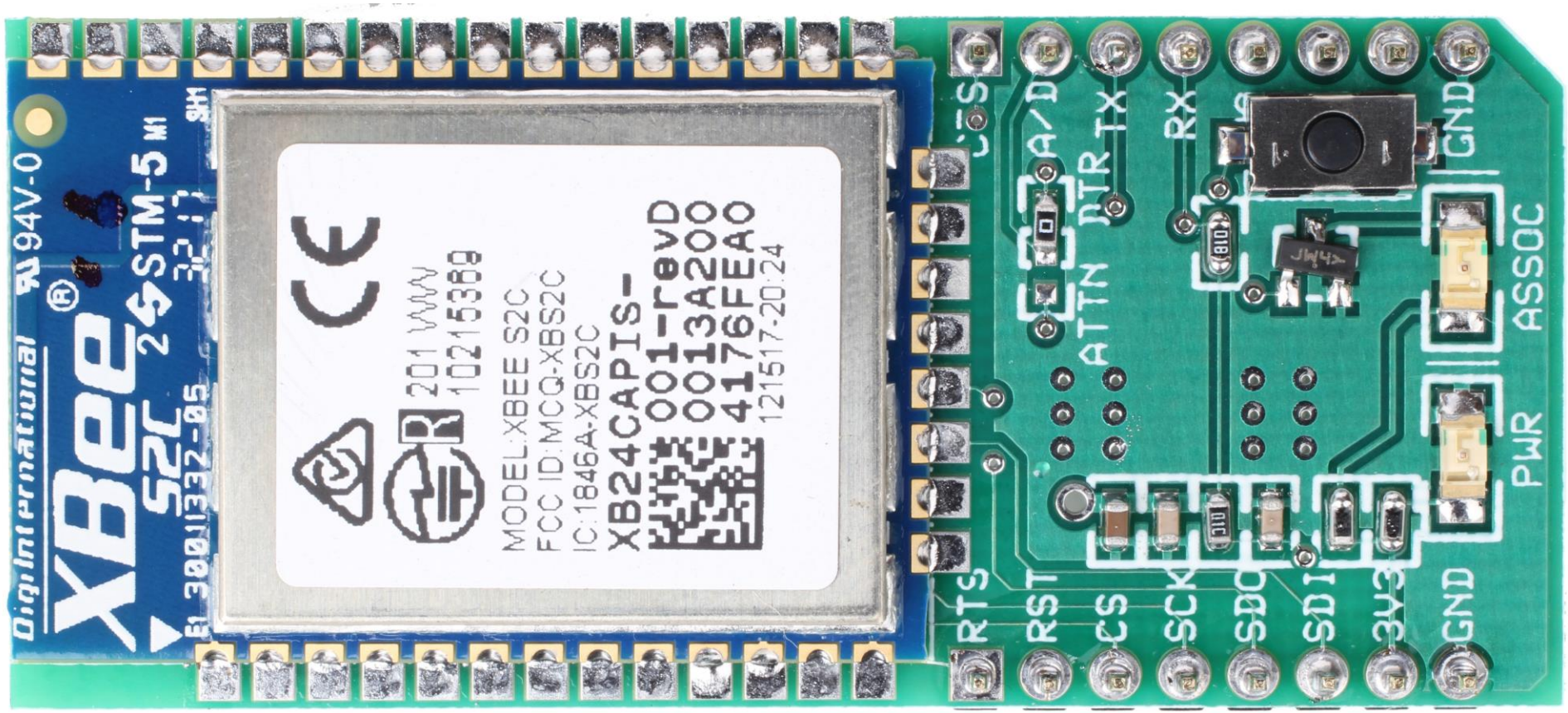
ARM Your Sensors

A Home-Brewed click-compatible XBee Module



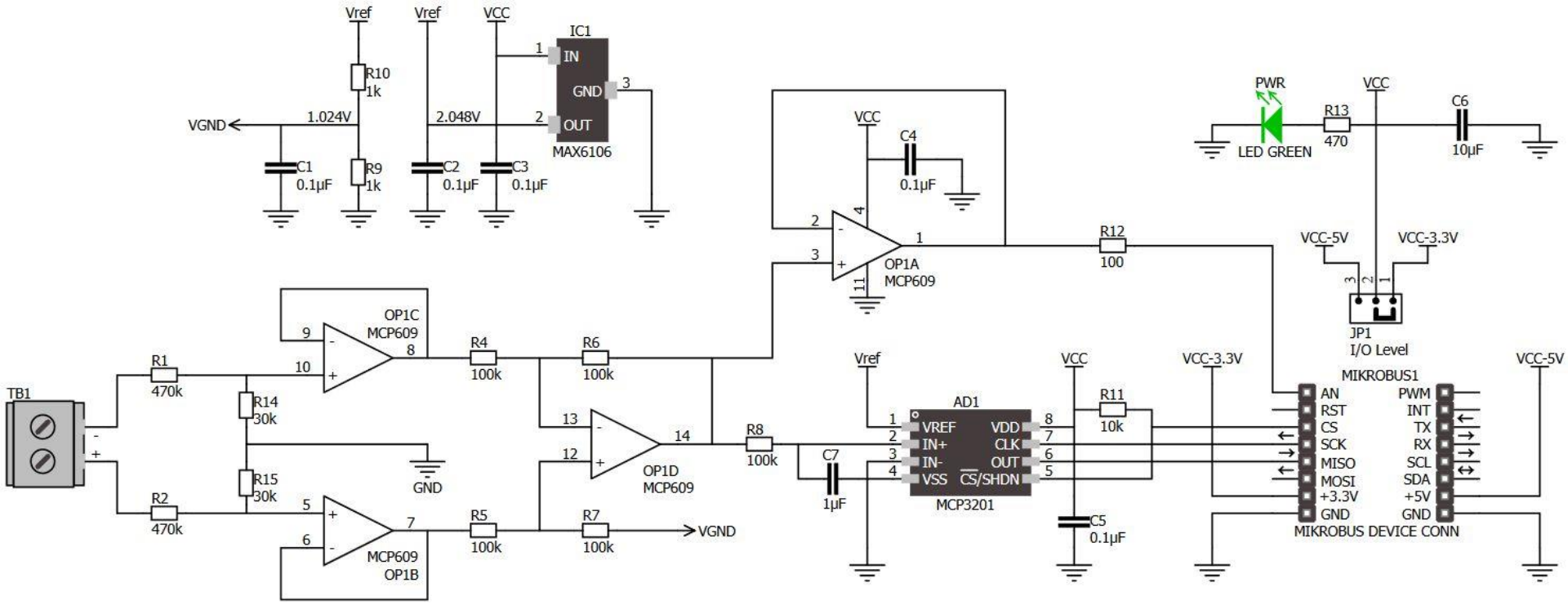
ARM Your Sensors

A Home-Brewed click-compatible XBee Module



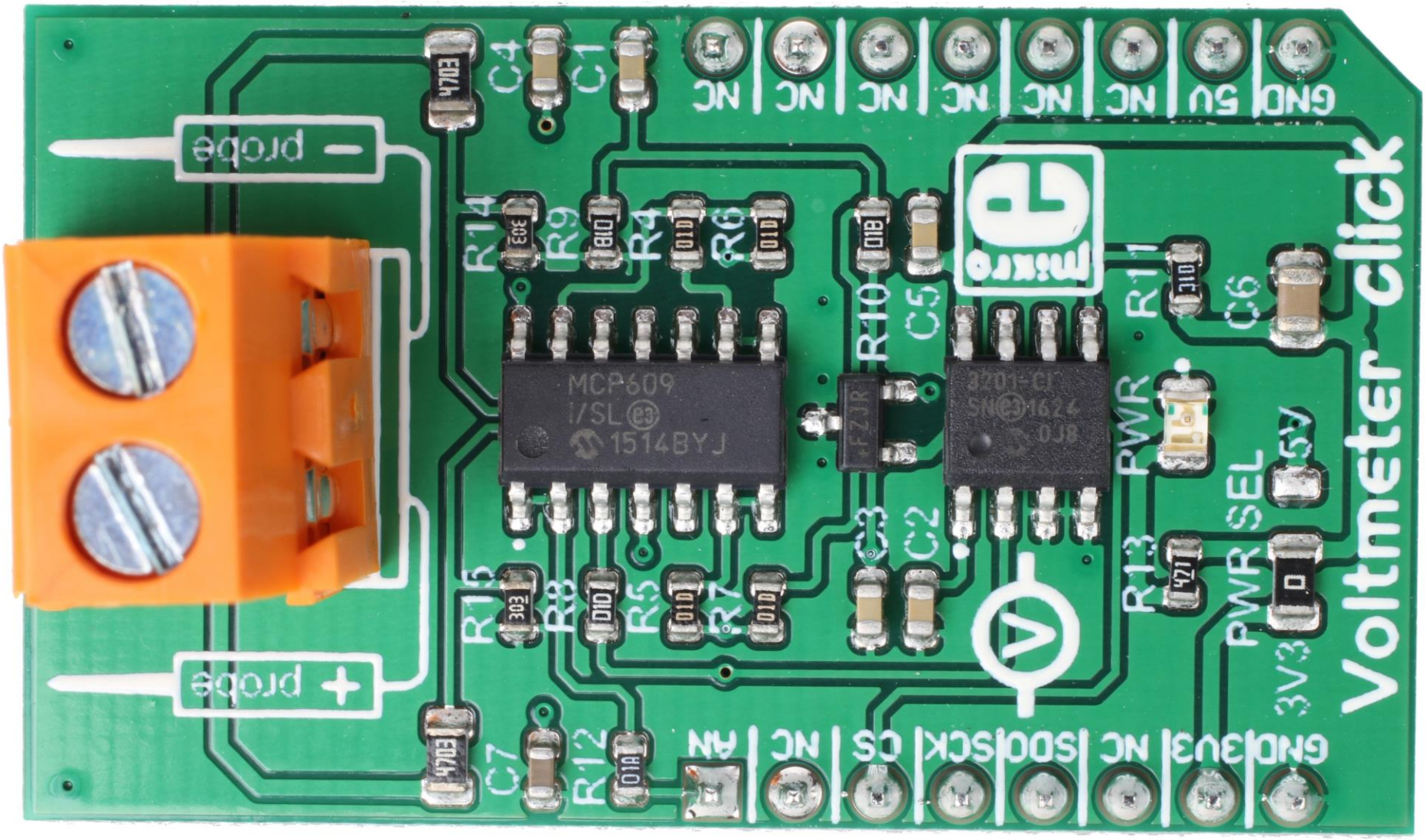
ARM Your Sensors

A Home-Brewed click-compatible XBee Module



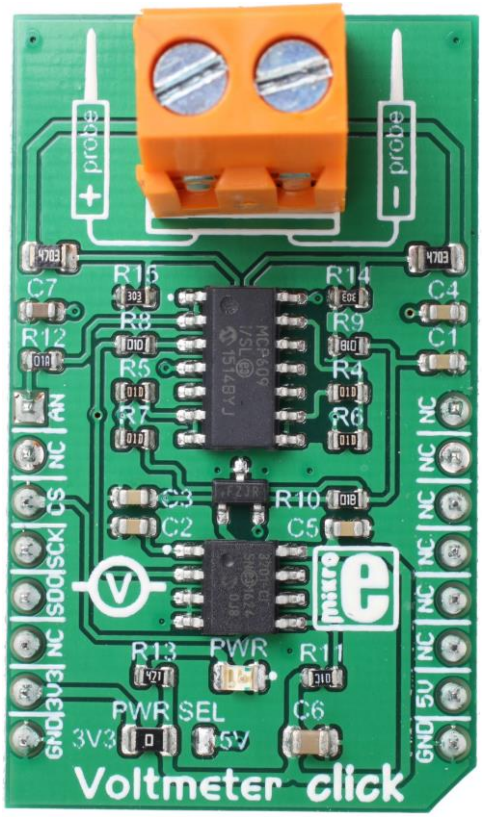
ARM Your Sensors

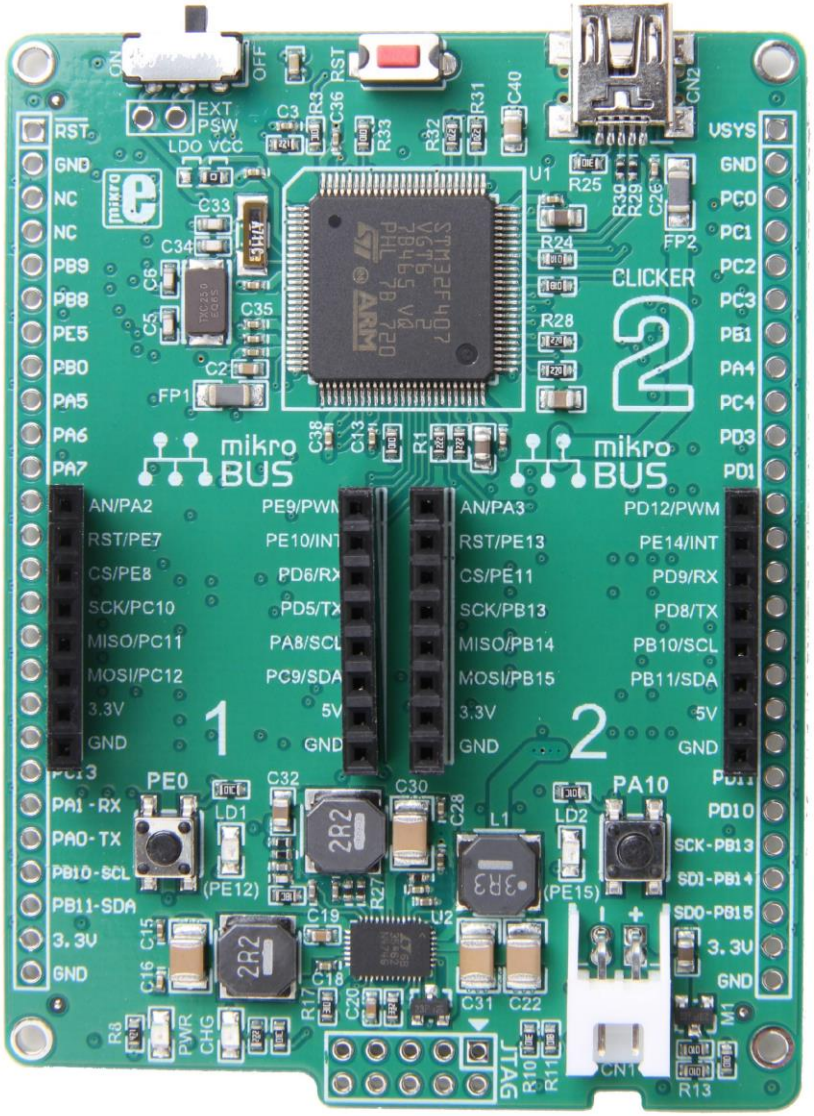
A Home-Brewed click-compatible XBee Module

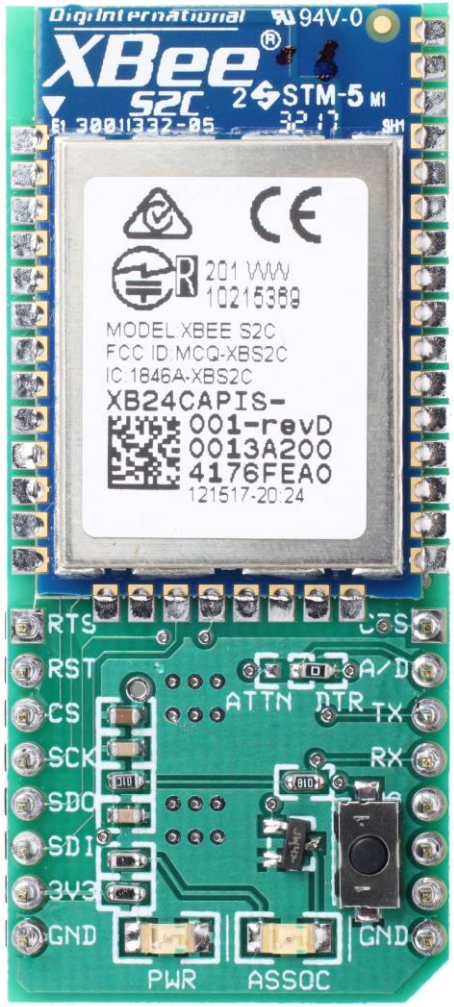


ARM Your Sensors

ARMing Our Sensor/XBee click-compatible Module

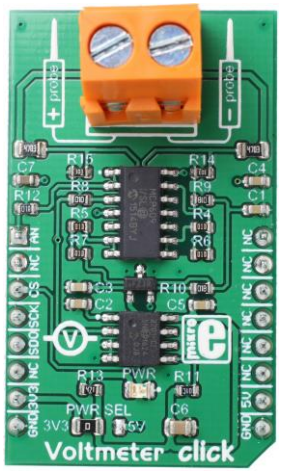






ARM Your Sensors

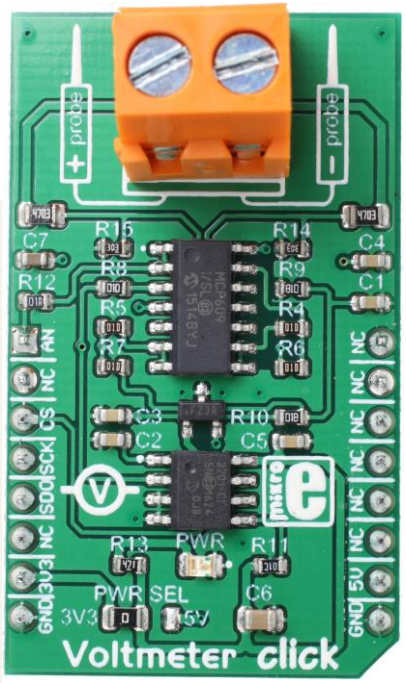
ARMing Our Sensor/XBee click-compatible Module




```
void system_init(void)
{
 GPIO_Digital_Output( &GPIOE_BASE, _GPIO_PINMASK_13 );      //XBee
 GPIO_Digital_Output( &GPIOE_BASE, _GPIO_PINMASK_8 );       //Voltmeter
 XBEE_RST = 1;
 METER_CS = 1;

 SPI3_Init_Advanced( _SPI_FPCLK_DIV16, _SPI_MASTER | _SPI_8_BIT |
                     _SPI_CLK_IDLE_LOW | _SPI_SECOND_CLK_EDGE_TRANSITION |
                     _SPI_MSB_FIRST | _SPI_SS_DISABLE | _SPI_SSM_ENABLE |
                     _SPI_SSI_1, &_GPIO_MODULE_SPI3_PC10_11_12 );
 UART3_Init_Advanced( 9600,
                      _UART_8_BIT_DATA,
                      _UART_NOPARITY,
                      _UART_ONE_STOPBIT,
                      &_GPIO_MODULE_USART3_PD89 );

 voltage = 0;
 sum = 0;
 measurement = 0;
 calibration = 0;

 calibration = getADC() / 2;
}
```

ARM Your Sensors

ARMing Our Sensor/XBee click-compatible Module




```
unsigned int getADC( void )
{
  char i, buffer;
  volatile unsigned int read, readl, avrg, sum;

  sum = 0;

  for (i = 0; i < 10; i++ )
  {
    METER_CS = 0;                    // Select MCP3201
    read = SPI3_Read(buffer);        // Get first 8 bits of ADC value
    readl = SPI3_Read(buffer);       // Get the rest of the ADC value bits
    METER_CS = 1;                    // Deselect MCP3201
    read = ((read & 0x1F) << 8);     // Store the first 8 bits of the ADC value in temporary variable
    read = ((read | readl) >> 1);    // Store the rest of the ADC value bits in temporary variable
    sum = sum + read;                // Sum of the eight ADC readings
  }
  avrg = sum / 10;                  // Average ADC value based on sum of the ADC readings
  return avrg;                      // Returns the average ADC value
}
```

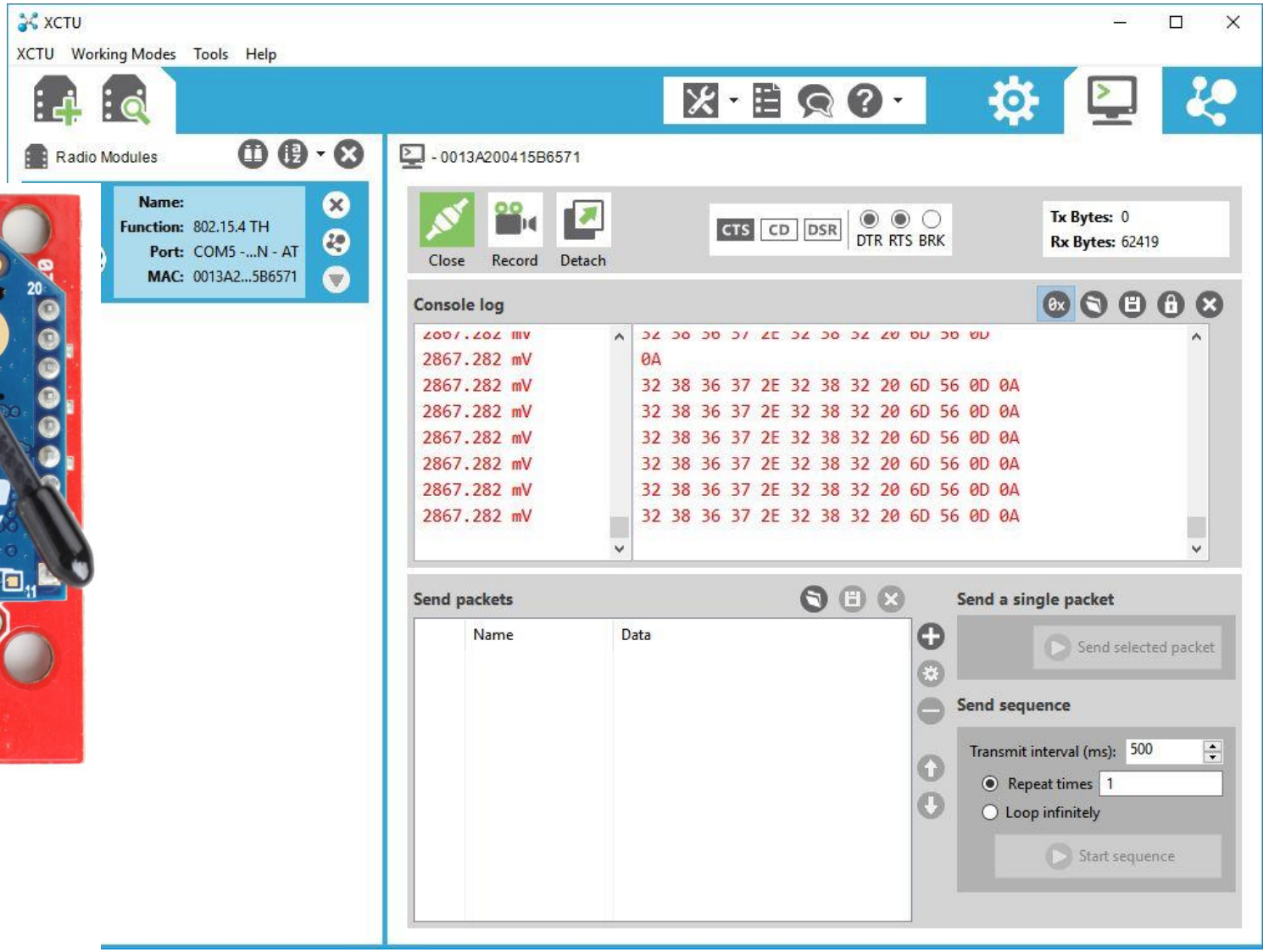
ARM Your Sensors

ARMing Our Sensor/XBee click-compatible Module

```
void main() {
     system_init();
    do{
       voltage = 0;

       measurement = getADC() / 2;
       voltage = (measurement - calibration) * 33.3405;

       FloatToStr(voltage, txt);

       UART3_Write_Text(txt);
       UART3_Write(32);
       UART3_Write_Text("mV");
       UART3_Write(13);
       UART_Write(10);

       delay_ms(1000);
     }while(1);
}
```

ARM Your Sensors

ARMing Our Sensor/XBee click-compatible Module



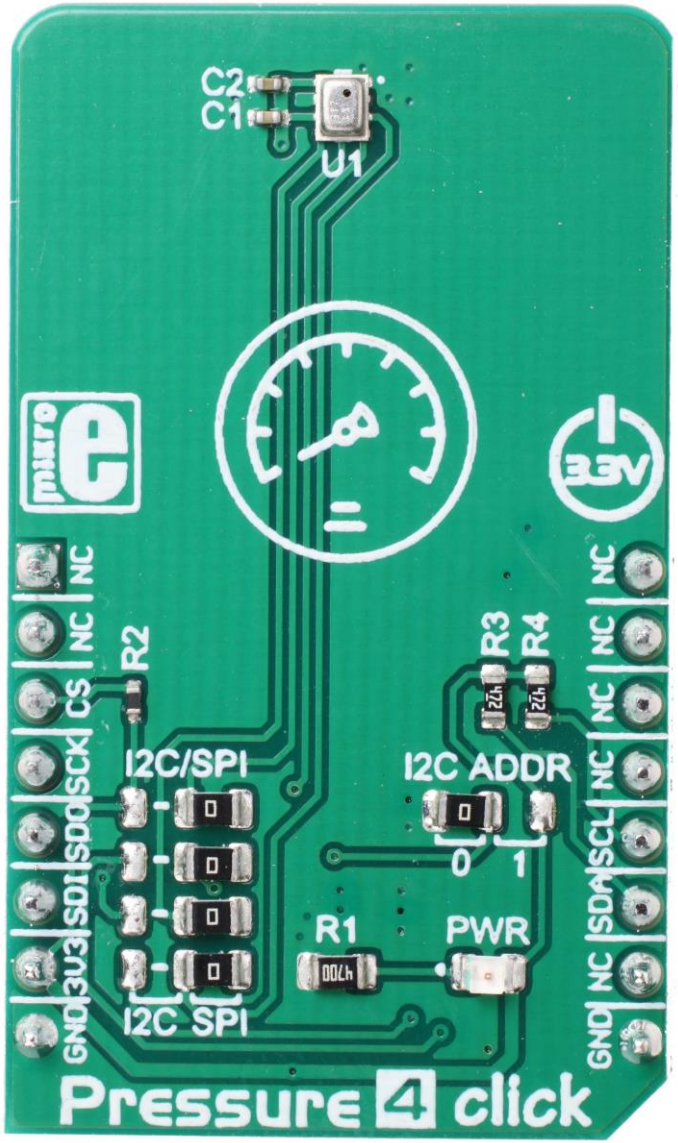
ARM Your Sensors

Under Pressure

```
#include "homeBrewXBee_types.h"

const uint32_t _HOME_BREW_XBEE_CFG[ 4 ] =
{
     9600,
    _UART_8_BIT_DATA,
    _UART_NOPARITY,
    _UART_ONE_STOPBIT
};
```

```
#ifndef _HBXBEE_T_
#define _HBXBEE_T_

#include "stdint.h"

#ifndef _HBXBEE_H_

#define T_HBXBEE_P const uint8_t*

#endif
#endif
```

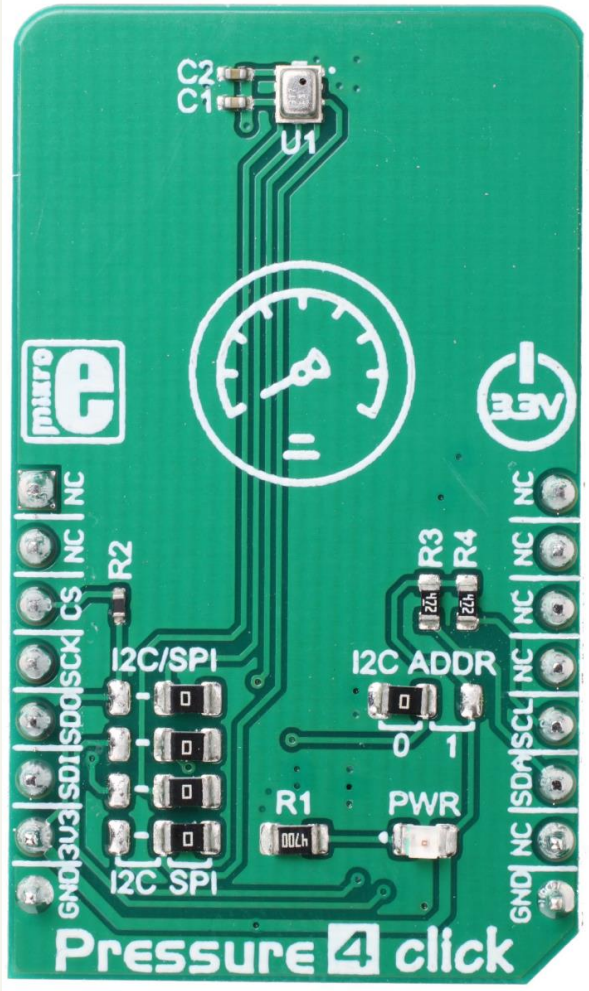
```
#include "Click_Pressure_4_types.h"
#include "Click_Pressure_4_config.h"
#include "homeBrewXBee_config.h"
#include "homeBrewXBee_types.h"

sbit XBEE_RST at GPIOE_ODR.B13;

void systemInit()
{
     //pressure click
    mikrobus_gpioInit( _MIKROBUS1, _MIKROBUS_CS_PIN, _GPIO_OUTPUT );
    mikrobus_spiInit( _MIKROBUS1, &_PRESSURE4_SPI_CFG[0] );
    //home brew xbee click
    mikrobus_gpioInit( _MIKROBUS2, _MIKROBUS_RST_PIN, _GPIO_OUTPUT );
    XBEE_RST =  1;
    mikrobus_uartInit( _MIKROBUS2, &_HOME_BREW_XBEE_CFG[0] );
    Delay_ms( 100 );
}
```



ARM Your Sensors

Under Pressure

```
void applicationInit()
{
    pressure4_spiDriverInit( (T_PRESSURE4_P)&_MIKROBUS1_GPIO, (T_PRESSURE4_P)&_MIKROBUS1_SPI );
    pressure4_init();
    Delay_ms(100);
}

void applicationTask()
{
    double tmp;
    char text[10];
    char scratch;

    UART3_Write_Text("Temperature : ");
    tmp = pressure4_getTemperature();
    FloatToStr(tmp,&text[0]);
    UART3_Write_Text(text);
    UART3_Write_Text(" C");
    UART3_Write(0x0D);
    UART3_Write(0x0A);

    UART3_Write_Text("Pressure : ");
    tmp = pressure4_getPressure();
    FloatToStr(tmp,&text[0]);
    UART3_Write_Text(text);
    UART3_Write_Text(" hPa");
    scratch = 2;
    do{
       UART3_Write(0x0D);
       UART3_Write(0x0A);
   }while(scratch--);

    Delay_ms(500);
}
```

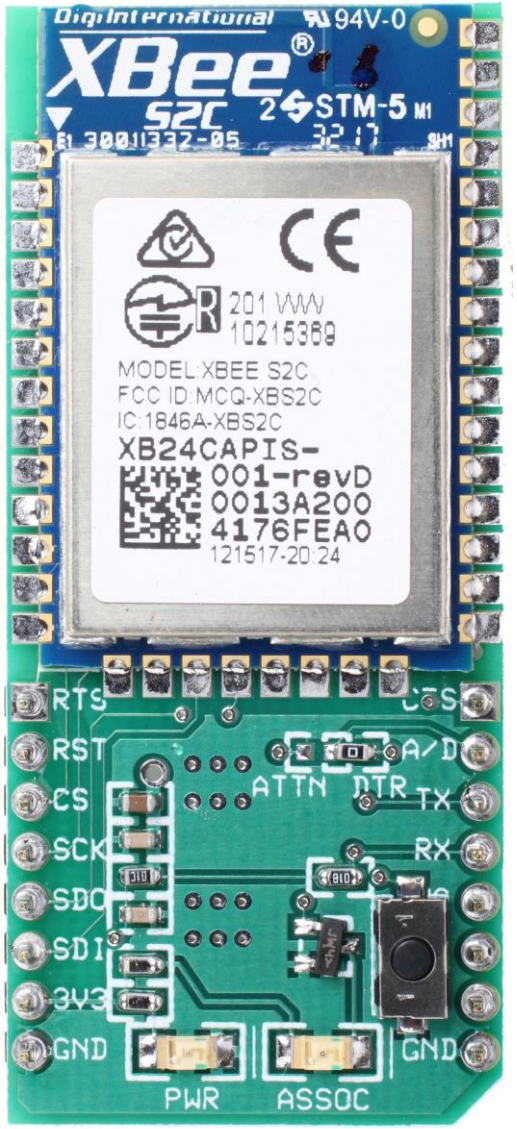


ARM Your Sensors

Under Pressure

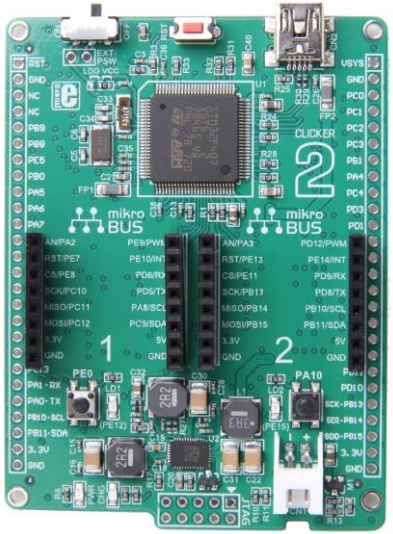

```
COM5 - Tera Term VT
File  Edit  Setup  Control  Window  Help
Pressure : 989.6195 hPa

Temperature : 27.38881 C
Pressure : 989.5656 hPa

Temperature : 27.39372 C
Pressure : 989.5869 hPa

Temperature : 27.39372 C
Pressure : 989.5887 hPa

Temperature : 27.39862 C
Pressure : 989.6148 hPa

Temperature : 27.38881 C
Pressure : 989.6172 hPa
```

```
void main()
{
    systemInit();
    applicationInit();

    while (1)
    {
        applicationTask();
    }
}
```



ARM Your Sensors

Day 4 Summary



mikroSDK