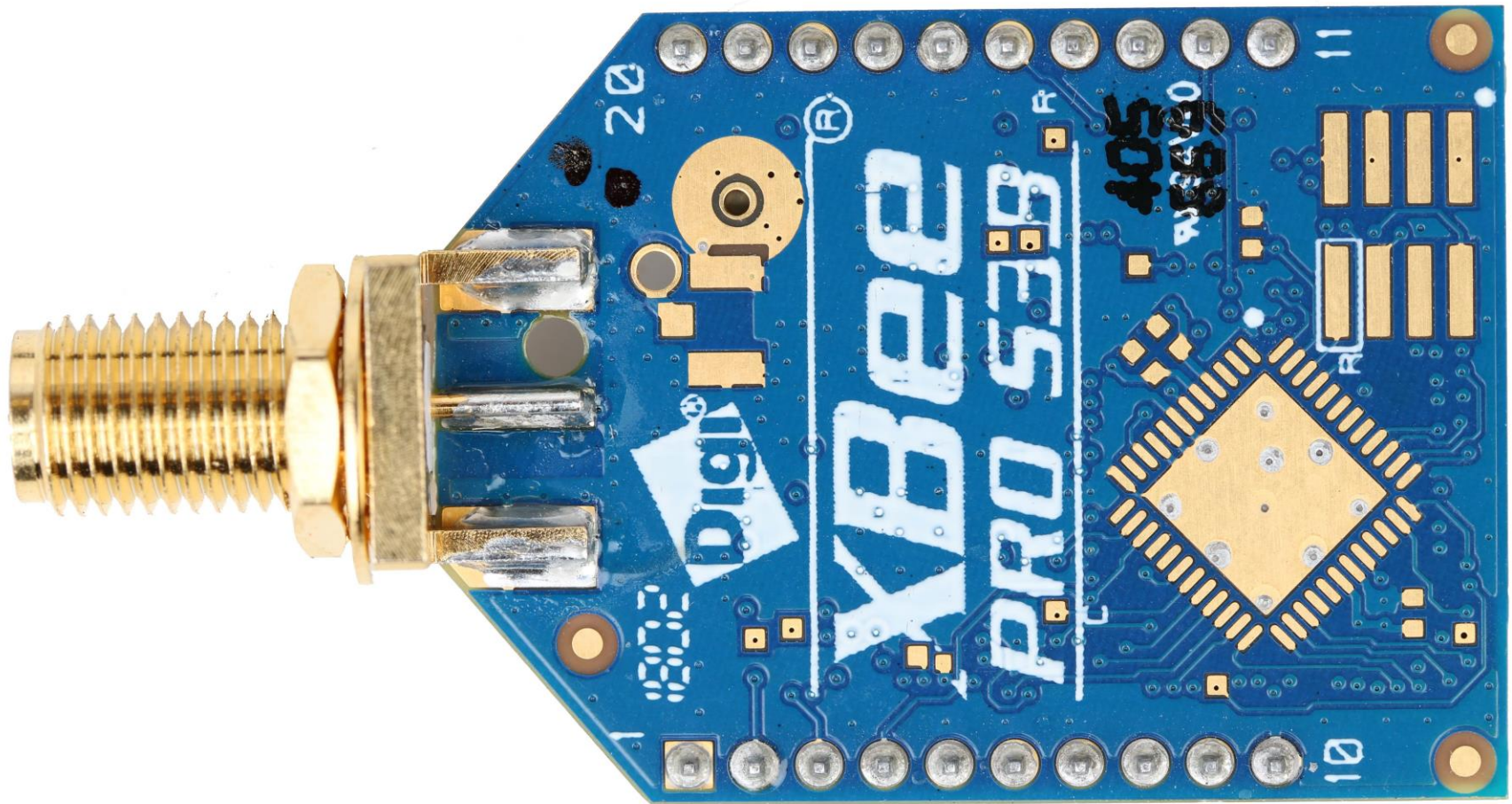


XBee Radio Modules



Building a Battery-Optimized High Power XBee Node

January 29, 2020

Fred Eady

Presented by:

DesignNews

Information Classification: General

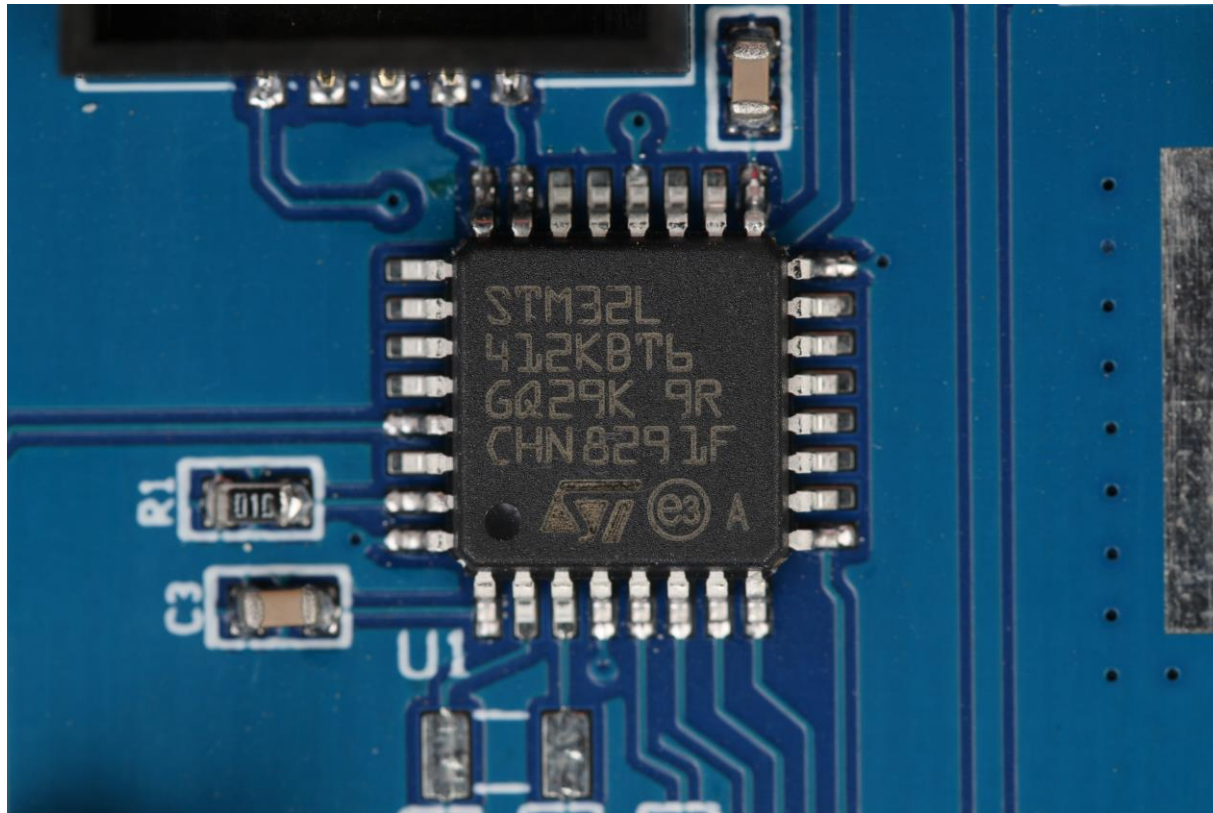
CEC CONTINUING
EDUCATION
CENTER



XBee Radio Modules

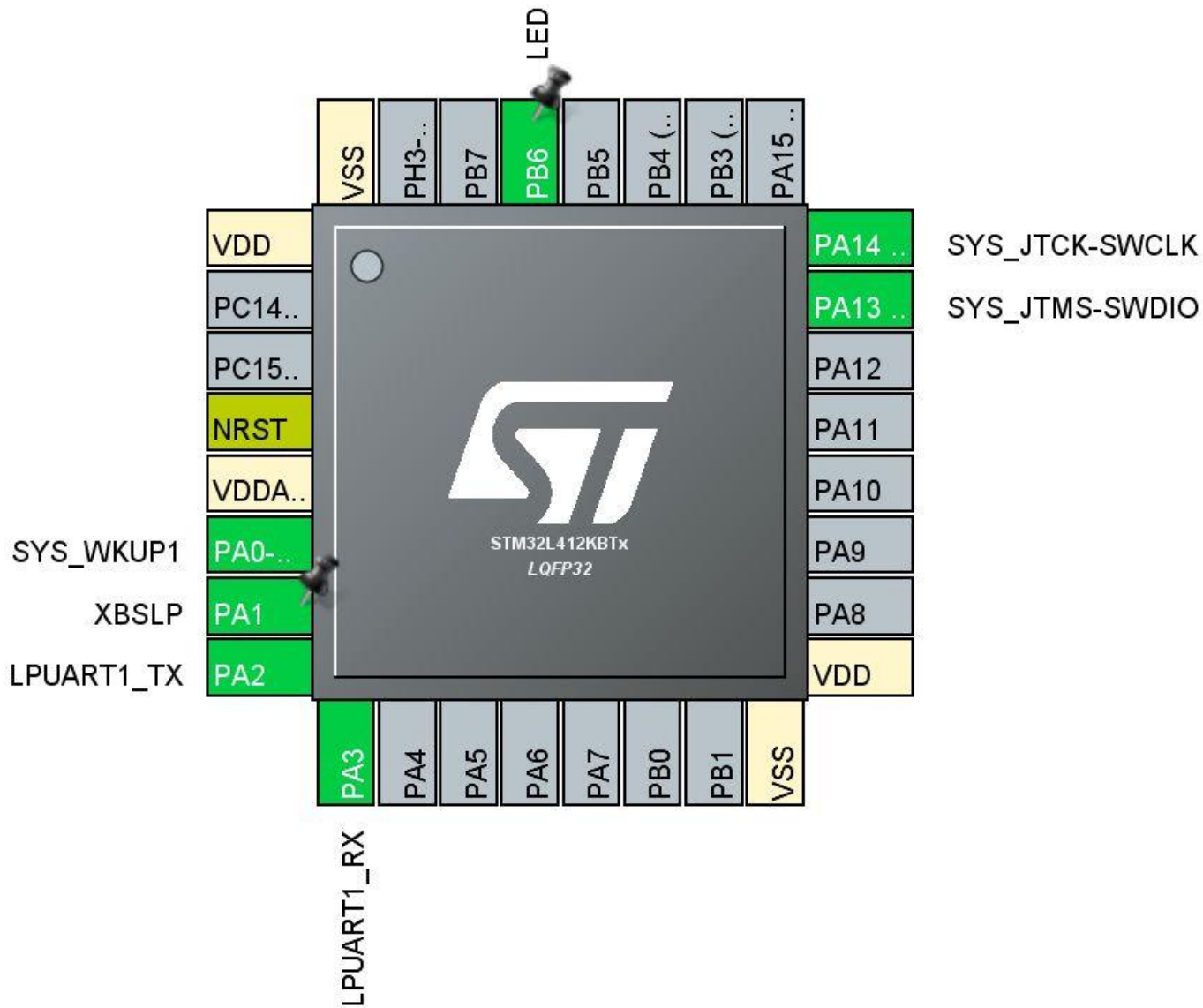
AGENDA

- XBee Hardware Design
- XBee Application Code
- Day 3 Summary



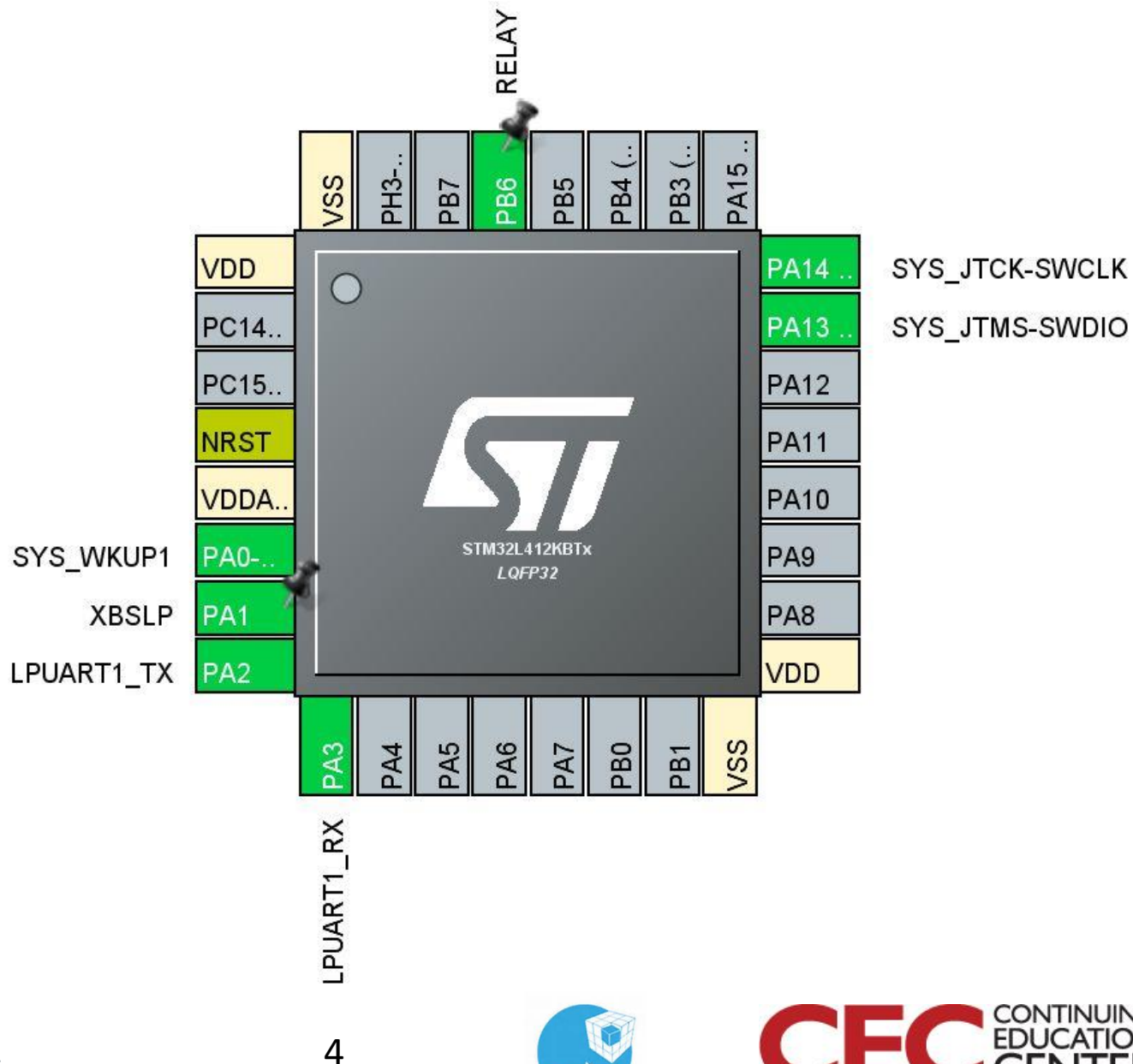
XBee Radio Modules

XBee Hardware Design - Transmitter Variant



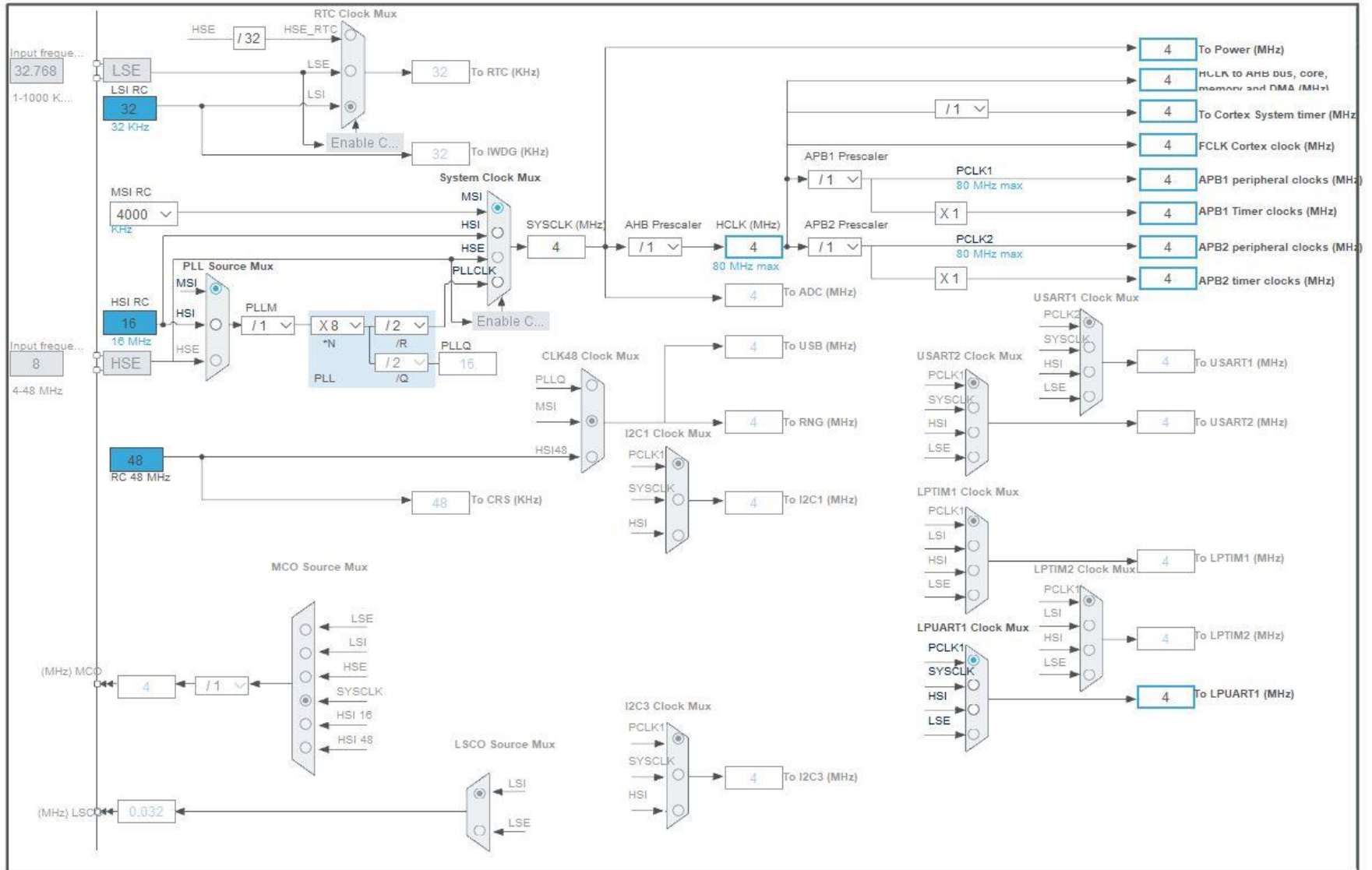
XBee Radio Modules

XBee Hardware Design - Receiver Variant



XBee Radio Modules

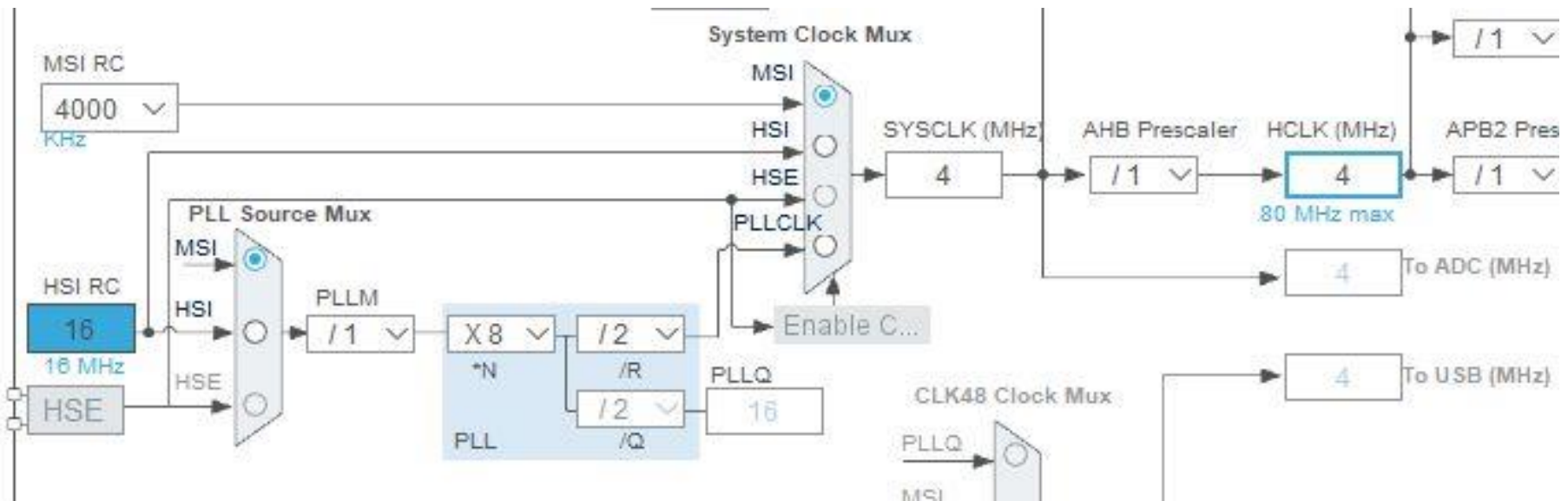
XBee Hardware Design - Transmitter/Receiver



Presented by:

XBee Radio Modules

XBee Hardware Design - Transmitter/Receiver



XBee Radio Modules

XBee Hardware Design - Transmitter/Receiver

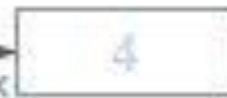
LPTIM1 Clock Mux



LPTIM2 Clock Mux



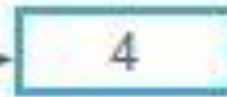
LPUART1 Clock Mux



To LPTIM1 (MHz)



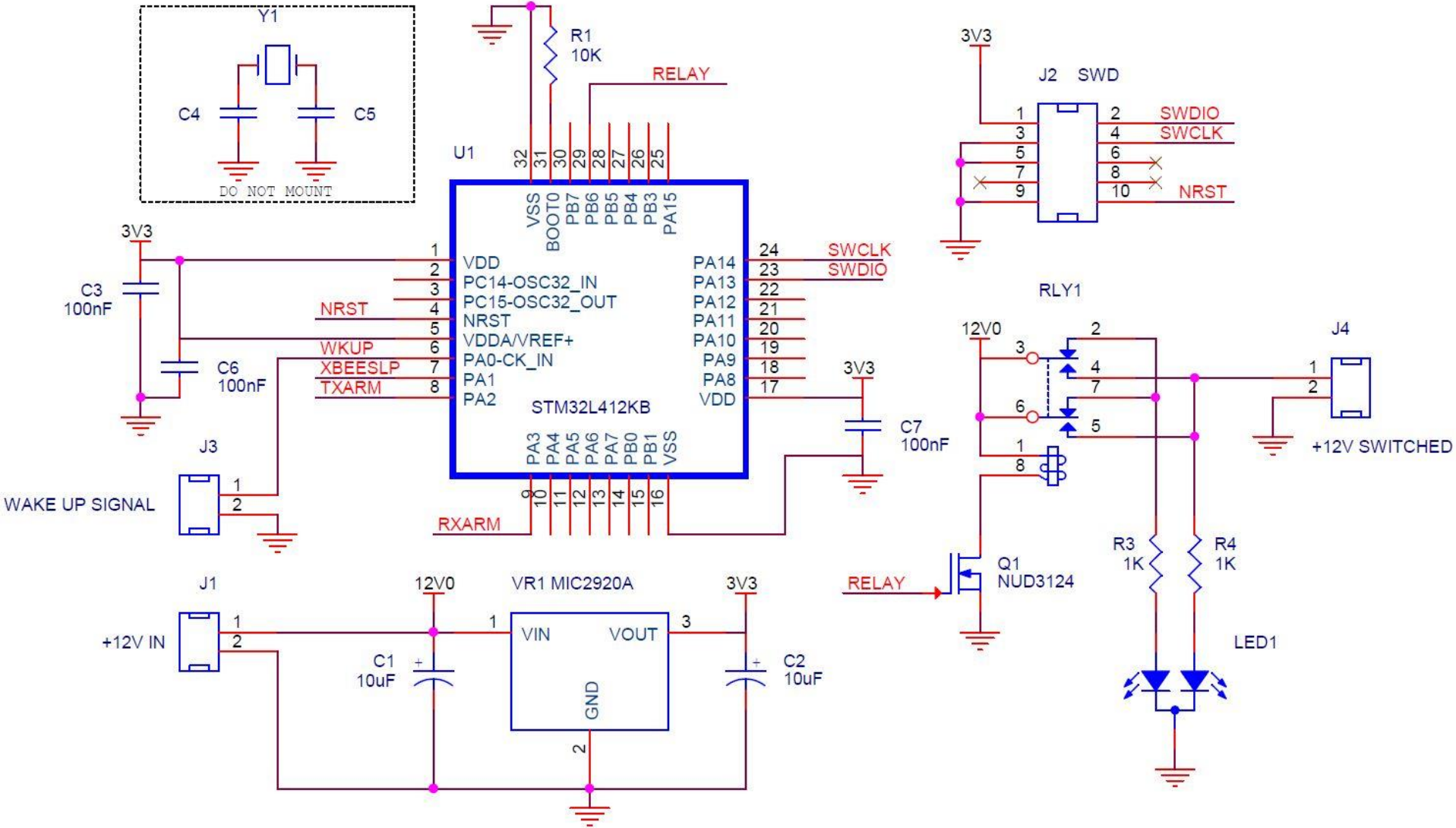
To LPTIM2 (MHz)



To LPUART1 (MHz)

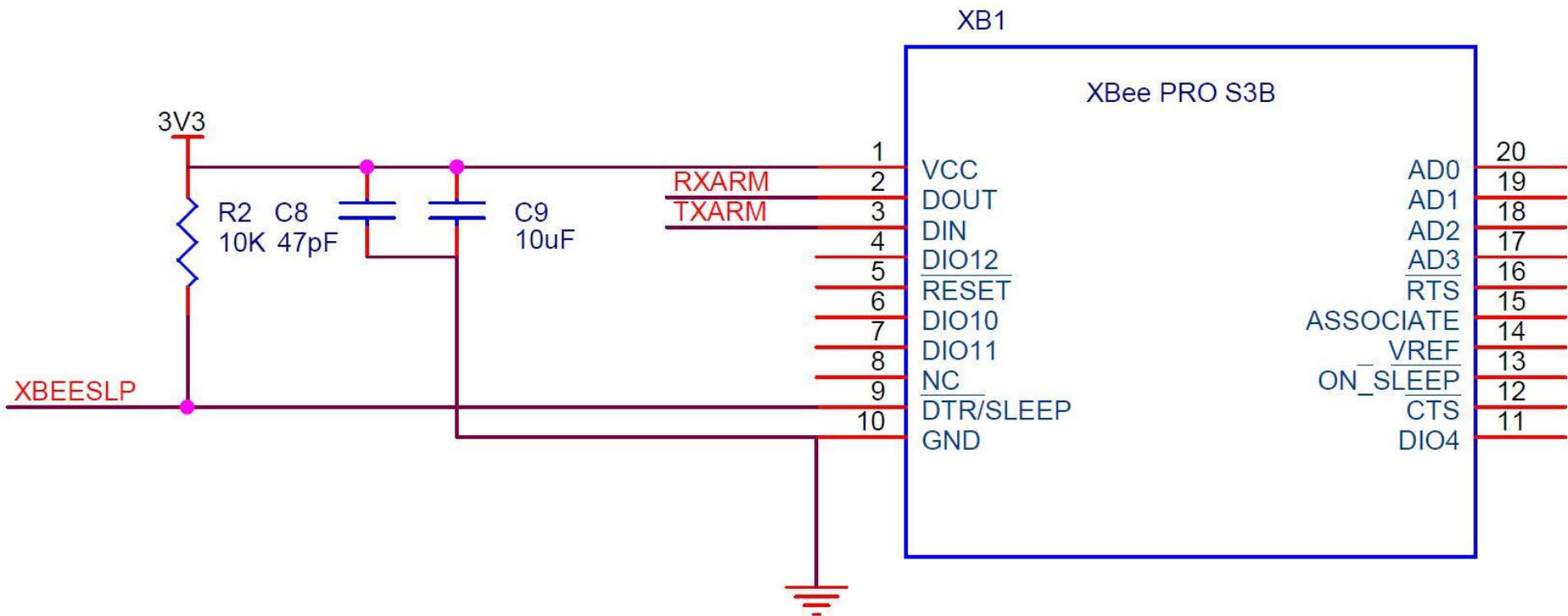
XBee Radio Modules

XBee Hardware Design - Transmitter/Receiver



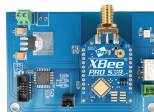
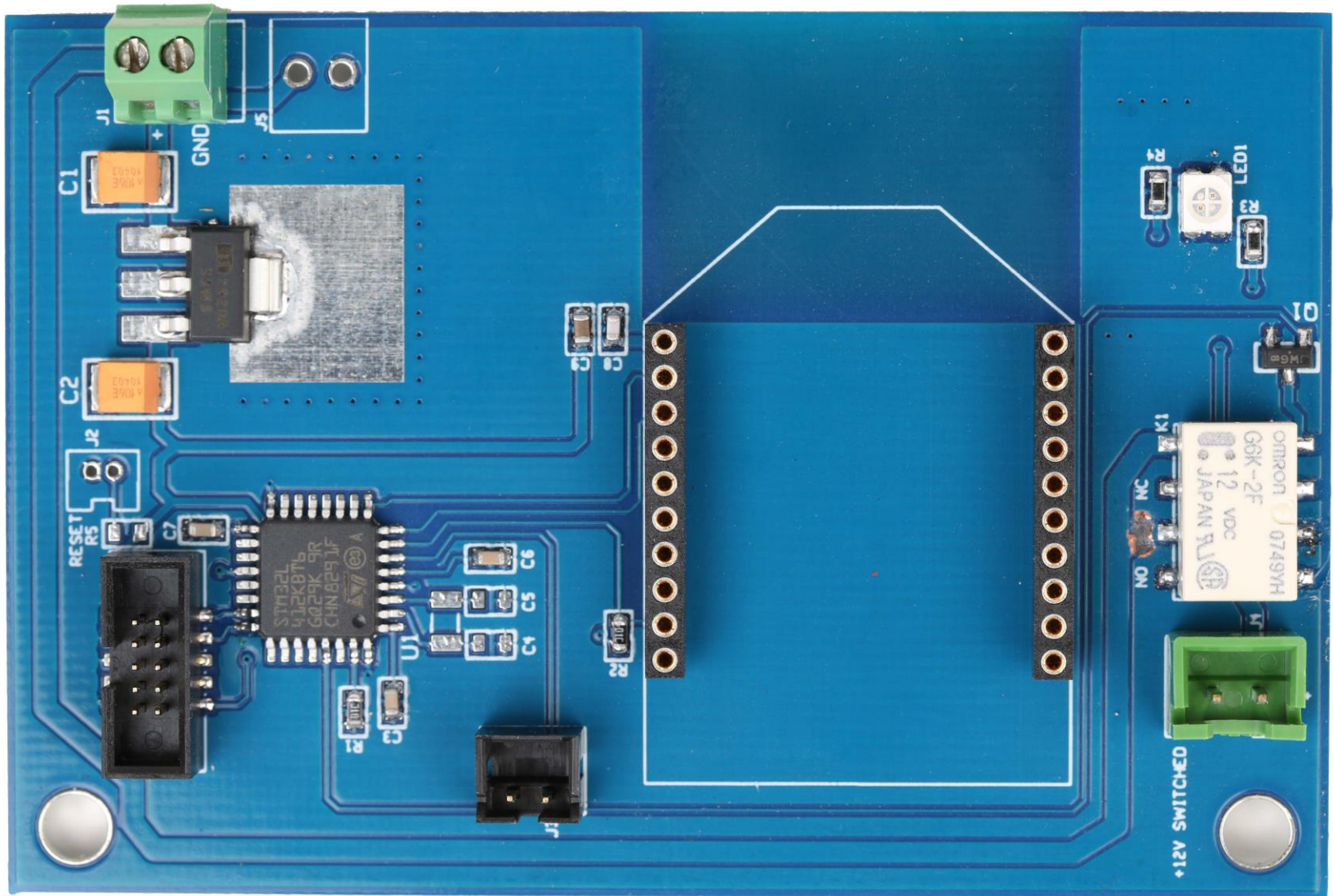
XBee Radio Modules

XBee Hardware Design - Transmitter/Receiver



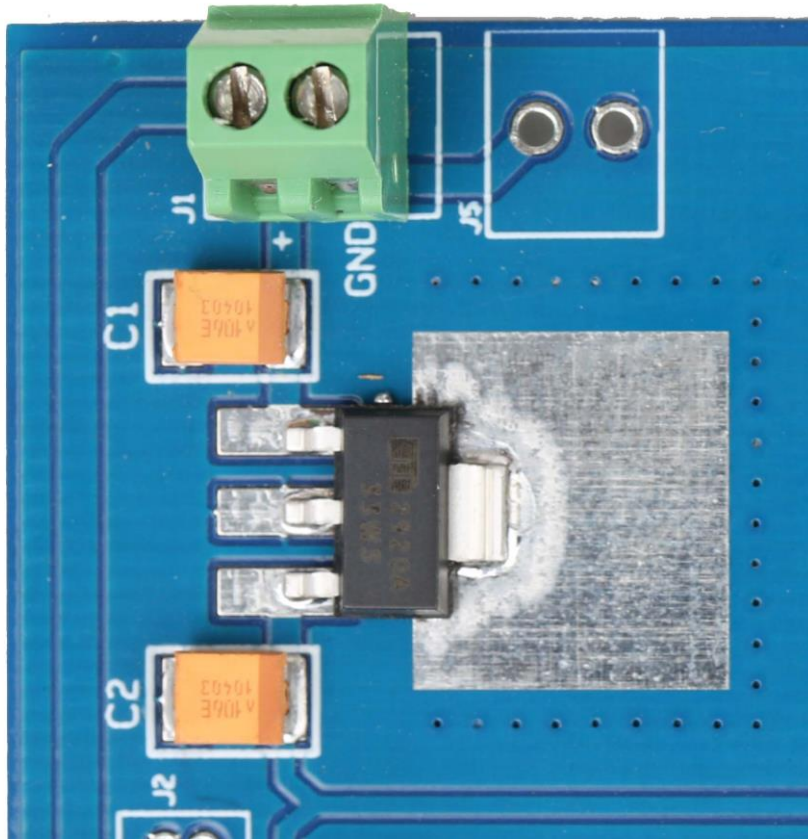
XBee Radio Modules

XBee Hardware Design - Transmitter/Receiver



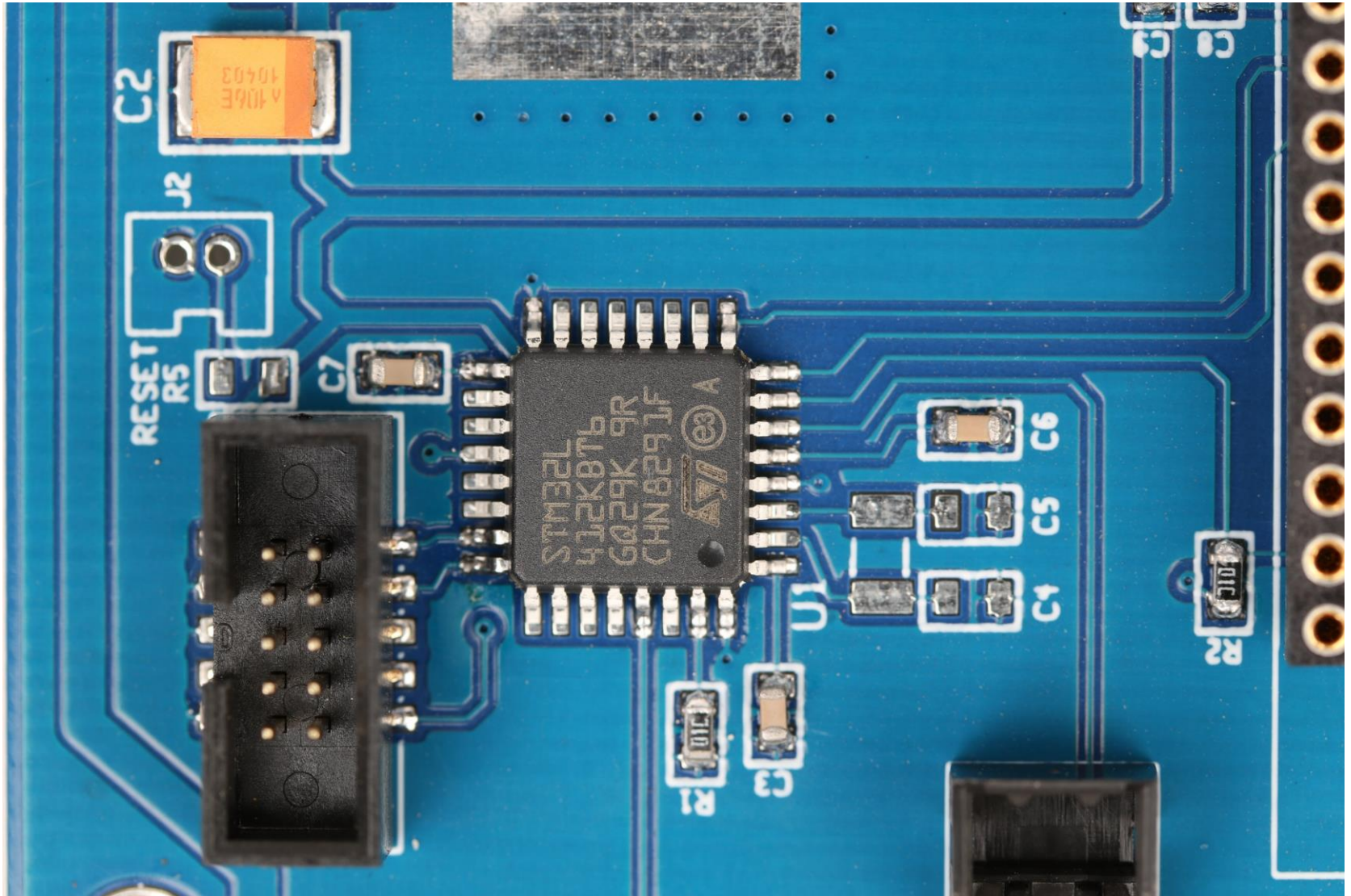
XBee Radio Modules

XBee Hardware Design - Transmitter/Receiver



XBee Radio Modules

XBee Hardware Design - Transmitter/Receiver

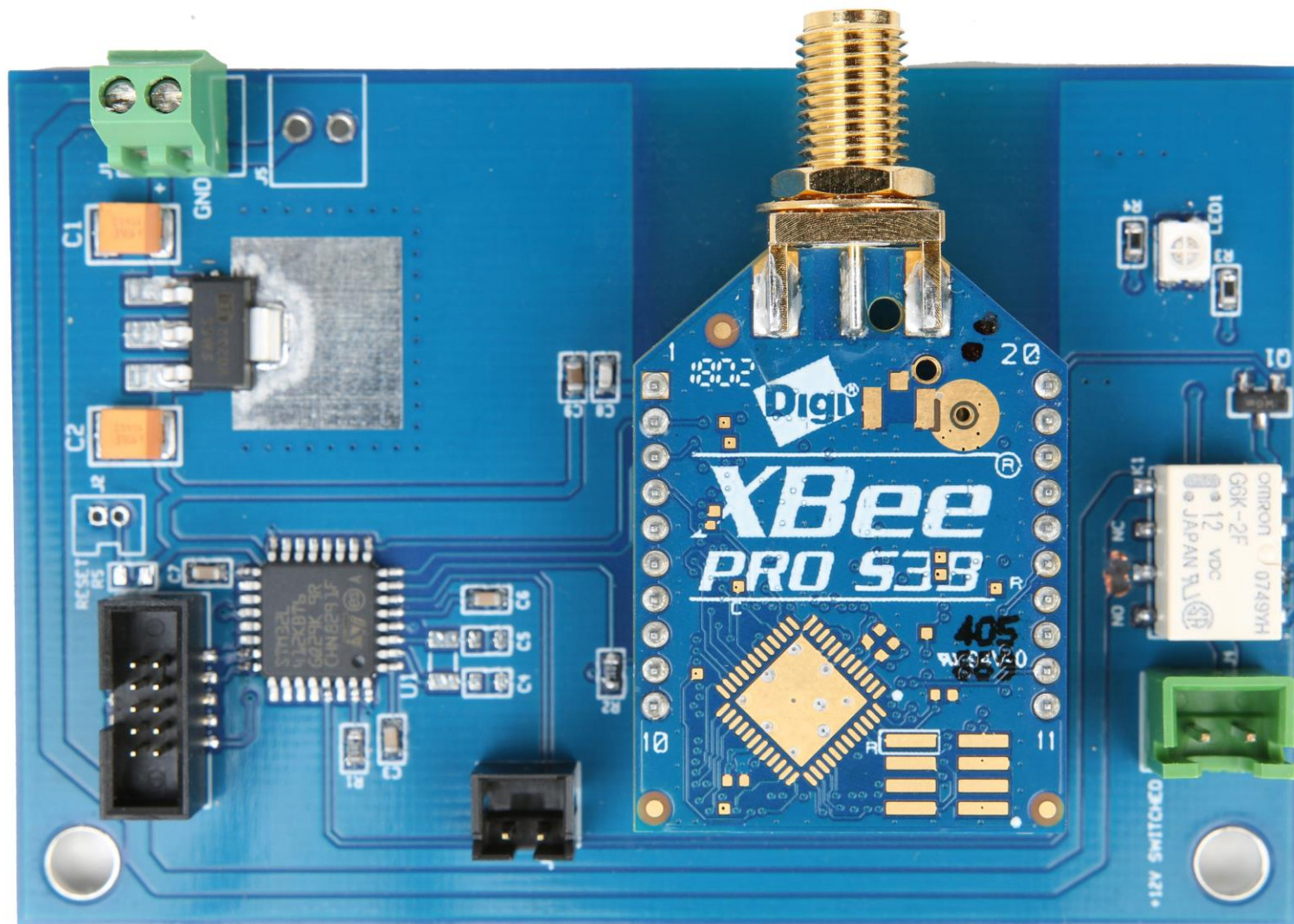


Presented by:



XBee Radio Modules

XBee Hardware Design - Transmitter/Receiver



XBee Radio Modules

XBee Hardware Design - Transmitter

XCTU Working Modes Tools Help

Radio Modules

Name:
Function: XBee PRO 900HP 10K
Port: COM8 - 9600/8/N/1/N - AT
MAC: 0013A2004182DA04

Radio Configuration [- 0013A2004182DA04]

Read Write Default Update Profile

Parameter

IR Sleep Sample rate

IR Sample Rate 0 * 1 ms

TP Temperature 1A

%V Supply Voltage CCC

▼ Sleep Commands
Configure Sleep Parameters

SM Sleep Mode Async. Pin Sleep [1]
Normal [0]

SO Sleep Options Async. Pin Sleep [1]
N/A [2]
N/A [3]

SN Number of Cycles On/Sleep

SP Sleep Time Async. Cyclic Sleep [4]
Async. Cyclic Sleep Pin Wake [5]

ST Wake Time BB8 * 1 ms

WH Wake Host Delay 0 * 1 ms

▼ AT Command Options
Change AT Command Mode Behavior

CC Command Sequence Character 2B Recomme...(ASCII)

CT Command Mode Timeout 64 * 100ms

GT Guard Times 3E8 * 1ms

▼ Firmware Version/Information

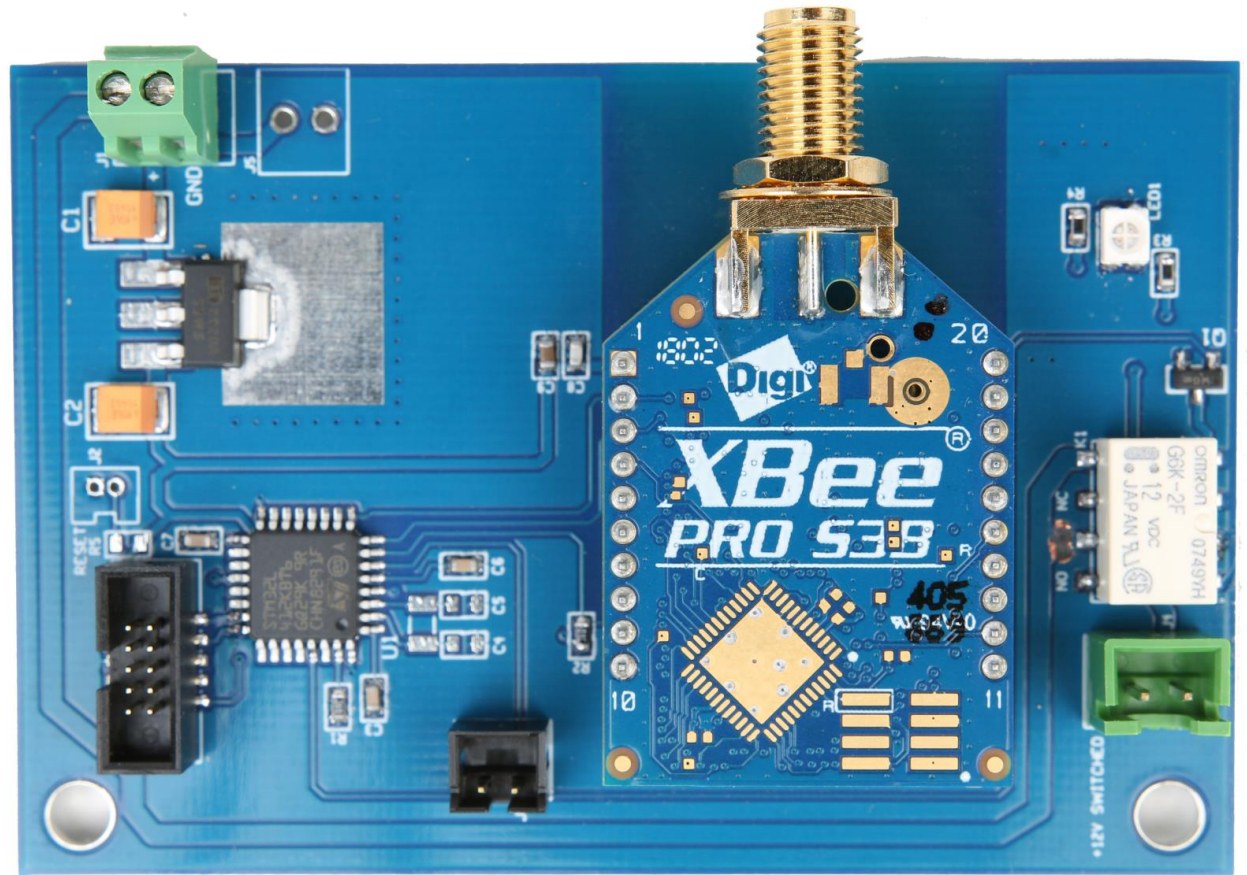
Presented by:



XBee Radio Modules

XBee Application Code

```
55 enum
56 {
57     TXALARM = 0,
58     MONITORSW,
59     TXALLCLR,
60     GOBACKTOSLEEP
61 };
62
```



XBee Radio Modules

XBee Application Code - GPIO Initialization

```
496 //*****
497 //*   NO RESTART BEGIN POINT
498 //*****
499 else
500 {
501     // Turn on LED
502     HAL_GPIO_WritePin(LED_GPIO_Port,LED_Pin,GPIO_PIN_SET);
503     // Insert 5 seconds delay
504     HAL_Delay(5000);
505     HAL_GPIO_WritePin(LED_GPIO_Port,LED_Pin,GPIO_PIN_RESET);
506     __HAL_RCC_GPIOA_CLK_ENABLE();
507     __HAL_RCC_GPIOB_CLK_ENABLE();
508     __HAL_RCC_GPIOC_CLK_ENABLE();
509     __HAL_RCC_GPIOH_CLK_ENABLE();
510
511     GPIO_InitStructure.Pin = GPIO_PIN_0;
512     GPIO_InitStructure.Mode = GPIO_MODE_INPUT;
513     GPIO_InitStructure.Pull = GPIO_PULLUP;
514     HAL_GPIO_Init(GPIOA, &GPIO_InitStructure);
515
516     //Configure GPIO pins : PC14 PC15
517     GPIO_InitStructure.Pin = GPIO_PIN_14|GPIO_PIN_15;
518     GPIO_InitStructure.Mode = GPIO_MODE_ANALOG;
519     GPIO_InitStructure.Pull = GPIO_NOPULL;
520     HAL_GPIO_Init(GPIOC, &GPIO_InitStructure);
521
522     //Configure GPIO pins : PA1 PA4 PA5 PA6 PA7 PA8 PA9 PA10 PA11 PA12 PA15
523     GPIO_InitStructure.Pin = GPIO_PIN_1|GPIO_PIN_4|GPIO_PIN_5|GPIO_PIN_6
524                             |GPIO_PIN_7|GPIO_PIN_8|GPIO_PIN_9|GPIO_PIN_10
525                             |GPIO_PIN_11|GPIO_PIN_12|GPIO_PIN_15;
526     GPIO_InitStructure.Mode = GPIO_MODE_ANALOG;
```


XBee Radio Modules

XBee Application Code - Power Down

```
543 // The Following Wakeup sequence is highly recommended prior to each Standby mode entry
544 //   mainly when using more than one wakeup source this is to not miss any wakeup event.
545 //   - Disable all used wakeup sources,
546 //   - Clear all related wakeup flags,
547 //   - Re-enable all used wakeup sources,
548 //   - Enter the Standby mode.
549
550
551 // For power consumption's sake, appropriately configure the GPIO corresponding to
552 //   the wake-up pin, fill up the pull-down control register and set the APC bit.
553
554 HAL_PWREx_EnableGPIOPullUp(PWR_GPIO_A, PWR_GPIO_BIT_0);
555 HAL_PWREx_EnablePullUpPullDownConfig();
556
557 // Disable used wakeup source: PWR_WAKEUP_PIN1
558 HAL_PWR_DisableWakeUpPin(PWR_WAKEUP_PIN1);
559
560 // Clear all related wakeup flags
561 __HAL_PWR_CLEAR_FLAG(PWR_FLAG_WU);
562
563 // Enable wakeup pin WKUP1
564 HAL_PWR_EnableWakeUpPin(PWR_WAKEUP_PIN1_LOW);
565
566 //Set TAMP back-up register TAMP_BKP31R to indicate
567 //later on that system has entered shutdown mode
568 WRITE_REG( TAMP->BKP31R, 0x01 );
569
570 // Enter the Shutdown mode
571 HAL_PWREx_EnterSHUTDOWNMode();
572
573 // This code will never be reached!
574 while (1)
575 {
576 }
577 }
```

XBee Radio Modules

XBee Application Code - Wake Up

```
184  __HAL_UART_ENABLE_IT(&hlpuart1,UART_IT_RXNE);
185
186  //Check if the system was resumed from shutdown mode,
187  //resort to TAMP back-up register TAMP_BKP31R to verify
188  //whether or not shutdown entry flag was set by software
189  //before entering shutdown mode.
190  if (READ_REG(TAMP->BKP31R) == 1)
191  {
192      WRITE_REG(TAMP->BKP31R, 0x00 ); /* reset back-up register */
193      // Blink LED to indicate that the system was resumed from Standby mode
194      HAL_GPIO_WritePin(LED_GPIO_Port,LED_Pin,GPIO_PIN_SET);
195      HAL_Delay(100);
196      HAL_GPIO_WritePin(LED_GPIO_Port,LED_Pin,GPIO_PIN_RESET);
197      //HAL_Delay(100);
198
199      __HAL_RCC_GPIOA_CLK_ENABLE();
200      __HAL_RCC_GPIOB_CLK_ENABLE();
201      __HAL_RCC_GPIOC_CLK_ENABLE();
202      __HAL_RCC_GPIOH_CLK_ENABLE();
203
204      GPIO_InitStructure.Pin = GPIO_PIN_0;
205      GPIO_InitStructure.Mode = GPIO_MODE_INPUT;
206      GPIO_InitStructure.Pull = GPIO_PULLUP;
207      HAL_GPIO_Init(GPIOA, &GPIO_InitStructure);
208
209      //Configure GPIO pins : PC14 PC15
210      GPIO_InitStructure.Pin = GPIO_PIN_14|GPIO_PIN_15;
211      GPIO_InitStructure.Mode = GPIO_MODE_ANALOG;
212      GPIO_InitStructure.Pull = GPIO_NOPULL;
213      HAL_GPIO_Init(GPIOC, &GPIO_InitStructure);
214
```

XBee Radio Modules

XBee Application Code - Send Alarm Signal

```
235 scratch8 = HAL_GPIO_ReadPin(GPIOA, GPIO_PIN_0) & 0x01;
236
237 if( scratch8 == 0)
238 {
239     // turn radion ON
240     LPUART1_RxHead = 0;
241     LPUART1_RxTail = 0;
242     GPIO_InitStructure.Pin = XBSLP_Pin;
243     GPIO_InitStructure.Mode = GPIO_MODE_OUTPUT_PP;
244     GPIO_InitStructure.Pull = GPIO_NOPULL;
245     HAL_GPIO_Init(GPIOA, &GPIO_InitStructure);
246     HAL_GPIO_WritePin(XBSLP_GPIO_Port,XBSLP_Pin,GPIO_PIN_RESET);
247     HAL_Delay(200);
248
249     txBuf[0] = 0xAA;
250     txBuf[1] = myAddr;
251     txBuf[2] = 0x41;//HAL_GPIO_ReadPin(GPIOA, GPIO_PIN_0) & 0x01;
252     txBuf[3] = 0xCC;
253     txBuf[4] = 0x33;
254     txBuf[5] = 0xC3;
255     txBuf[6] = 0x3C;
256     HAL_UART_Transmit(&hlpuart1, txBuf, 7, 0xFFFF);
257     HAL_Delay(100);
258
259     txLoopOuter = outerLoopVal;
260     txLoopInner = innerLoopVal;
261     flags.fflop = 1;
262     pstate = TXALARM;
```

XBee Radio Modules

XBee Application Code - Send Alarm Until Acknowledged

```
270     case TXALARM:
271         if(txLoopOuter-- == 0)
272         {
273             if(txLoopInner-- == 0)
274             {
275                 txBuf[0] = 0xAA;
276                 txBuf[1] = myAddr;
277                 txBuf[2] = 0x41;
278                 txBuf[3] = 0xCC;
279                 txBuf[4] = 0x33;
280                 txBuf[5] = 0xC3;
281                 txBuf[6] = 0x3C;
282                 HAL_UART_Transmit(&hlpuart1, txBuf, 7, 0xFFFF);
283                 HAL_GPIO_WritePin(LED_GPIO_Port, LED_Pin, GPIO_PIN_SET);
284                 HAL_Delay(100);
285                 HAL_GPIO_WritePin(LED_GPIO_Port, LED_Pin, GPIO_PIN_RESET);
286
287                 if(flags.fflop == 1)
288                 {
289                     txLoopOuter = outerLoopVal;
290                     txLoopInner = innerLoopVal - 2;
291                     flags.fflop = 0;
292                 }
293                 else
294                 {
295                     txLoopOuter = outerLoopVal;
296                     txLoopInner = innerLoopVal;
297                     flags.fflop = 1;
298                 }
299             }
300         }
301         else
302         {
303             txLoopOuter = outerLoopVal;
304         }
```

XBee Radio Modules

XBee Application Code - Alarm Acknowledged

```
306     if(CharInRing())
307     {
308         rxBuf[0] = 0x00;
309         rxBuf[0] = readring();
310         if(rxBuf[0] == 0x55)
311         {
312             HAL_Delay(200);
313             rxBuf[1] = readring(); //my addr
314             rxBuf[2] = readring(); //0xCC
315             rxBuf[3] = readring(); //0x33
316             rxBuf[4] = readring(); //0xC3
317             rxBuf[5] = readring(); //0x3C
318
319             if(rxBuf[1] == myAddr &&
320                rxBuf[2] == 0xCC &&
321                rxBuf[3] == 0x33 &&
322                rxBuf[4] == 0xC3 &&
323                rxBuf[5] == 0x3C)
324             {
325                 // turn radio OFF
326                 GPIO_InitStructure.Pin = XBSLP_Pin;
327                 GPIO_InitStructure.Mode = GPIO_MODE_ANALOG;
328                 GPIO_InitStructure.Pull = GPIO_NOPULL;
329                 HAL_GPIO_Init(GPIOA, &GPIO_InitStructure);
330                 pstate = MONITORSW;
331             }
332         }
333     }
334     break;
```

XBee Radio Modules

XBee Application Code - Monitor Alarm Sensor

```
338     case MONITORSW:
339         blbStatus = HAL_GPIO_ReadPin(GPIOA, GPIO_PIN_0) & 0x01;
340         if(blbStatus == 0x01)
341         {
342             HAL_Delay(100);
343             blbStatus = HAL_GPIO_ReadPin(GPIOA, GPIO_PIN_0) & 0x01;
344             if(blbStatus == 0x01)
345             {
346                 txLoopOuter = outerLoopVal;
347                 txLoopInner = innerLoopVal;
348                 flags.fflop = 1;
349                 // turn radion ON
350                 LPUART1_RxHead = 0;
351                 LPUART1_RxTail = 0;
352                 GPIO_InitStructure.Pin = XBSLP_Pin;
353                 GPIO_InitStructure.Mode = GPIO_MODE_OUTPUT_PP;
354                 GPIO_InitStructure.Pull = GPIO_NOPULL;
355                 HAL_GPIO_Init(GPIOA, &GPIO_InitStructure);
356                 HAL_GPIO_WritePin(XBSLP_GPIO_Port,XBSLP_Pin,GPIO_PIN_RESET);
357                 HAL_Delay(200);
358                 txBuf[0] = 0xAA;
359                 txBuf[1] = myAddr;
360                 txBuf[2] = 0x43;
361                 txBuf[3] = 0xCC;
362                 txBuf[4] = 0x33;
363                 txBuf[5] = 0xC3;
364                 txBuf[6] = 0x3C;
365                 HAL_UART_Transmit(&hlpuart1, txBuf, 7, 0xFFFF);
366                 HAL_GPIO_WritePin(LED_GPIO_Port,LED_Pin,GPIO_PIN_SET);
367                 HAL_Delay(100);
368                 HAL_GPIO_WritePin(LED_GPIO_Port,LED_Pin,GPIO_PIN_RESET);
369                 pstate = TXALLCLR;
370             }
371         }
372     break;
```

XBee Radio Modules

XBee Application Code - Send All Clear

```
377     case TXALLCLR:
378         if(txLoopOuter-- == 0)
379         {
380             if(txLoopInner-- == 0)
381             {
382                 txBuf[0] = 0xAA;
383                 txBuf[1] = myAddr;
384                 txBuf[2] = 0x43;
385                 txBuf[3] = 0xCC;
386                 txBuf[4] = 0x33;
387                 txBuf[5] = 0xC3;
388                 txBuf[6] = 0x3C;
389                 HAL_UART_Transmit(&hlpuart1, txBuf, 7, 0xFFFF);
390                 HAL_GPIO_WritePin(LED_GPIO_Port, LED_Pin, GPIO_PIN_SET);
391                 HAL_Delay(100);
392                 HAL_GPIO_WritePin(LED_GPIO_Port, LED_Pin, GPIO_PIN_RESET);
393
394                 if(flags.fflop == 1)
395                 {
396                     txLoopOuter = outerLoopVal;
397                     txLoopInner = innerLoopVal - 2;
398                     flags.fflop = 0;
399                 }
400                 else
401                 {
402                     txLoopOuter = outerLoopVal;
403                     txLoopInner = innerLoopVal;
404                     flags.fflop = 1;
405                 }
406             }
407         }
408         else
409         {
410             txLoopOuter = outerLoopVal;
411         }
412     }
```

XBee Radio Modules

XBee Application Code - All Clear Acknowledged - Go Back To Sleep

```
441     case GOBACKTOSLEEP:
442         // turn radio OFF
443         GPIO_InitStructure.Pin = XBSLP_Pin;
444         GPIO_InitStructure.Mode = GPIO_MODE_ANALOG;
445         GPIO_InitStructure.Pull = GPIO_NOPULL;
446         HAL_GPIO_Init(GPIOA, &GPIO_InitStructure);
447
448         for(scratch8=0;scratch8<3;scratch8++)
449         {
450             HAL_GPIO_WritePin(LED_GPIO_Port,LED_Pin,GPIO_PIN_SET);
451             HAL_Delay(25);
452             HAL_GPIO_WritePin(LED_GPIO_Port,LED_Pin,GPIO_PIN_RESET);
453             HAL_Delay(100);
454         }
455
456         GPIO_InitStructure.Pin = LED_Pin;
457         GPIO_InitStructure.Mode = GPIO_MODE_ANALOG;
458         GPIO_InitStructure.Pull = GPIO_NOPULL;
459         HAL_GPIO_Init(GPIOB, &GPIO_InitStructure);
```


XBee Radio Modules

XBee Application Code - All Clear Acknowledged - Go Back To Sleep

```
473 HAL_PWREx_EnableGPIOPullUp(PWR_GPIO_A, PWR_GPIO_BIT_0);
474 HAL_PWREx_EnablePullUpPullDownConfig();
475
476 // Disable used wakeup source: PWR_WAKEUP_PIN1
477 HAL_PWR_DisableWakeUpPin(PWR_WAKEUP_PIN1);
478
479 // Clear all related wakeup flags
480 __HAL_PWR_CLEAR_FLAG(PWR_FLAG_WU);
481
482 // Enable wakeup pin WKUP1
483 HAL_PWR_EnableWakeUpPin(PWR_WAKEUP_PIN1_LOW);
484
485 //Set TAMP back-up register TAMP_BKP31R to indicate
486 //later on that system has entered shutdown mode
487 WRITE_REG( TAMP->BKP31R, 0x01 );
488
489 // Enter the Shutdown mode
490 HAL_PWREx_EnterSHUTDOWNMode();
491 break;
```

XBee Radio Modules

Day 3 Summary

