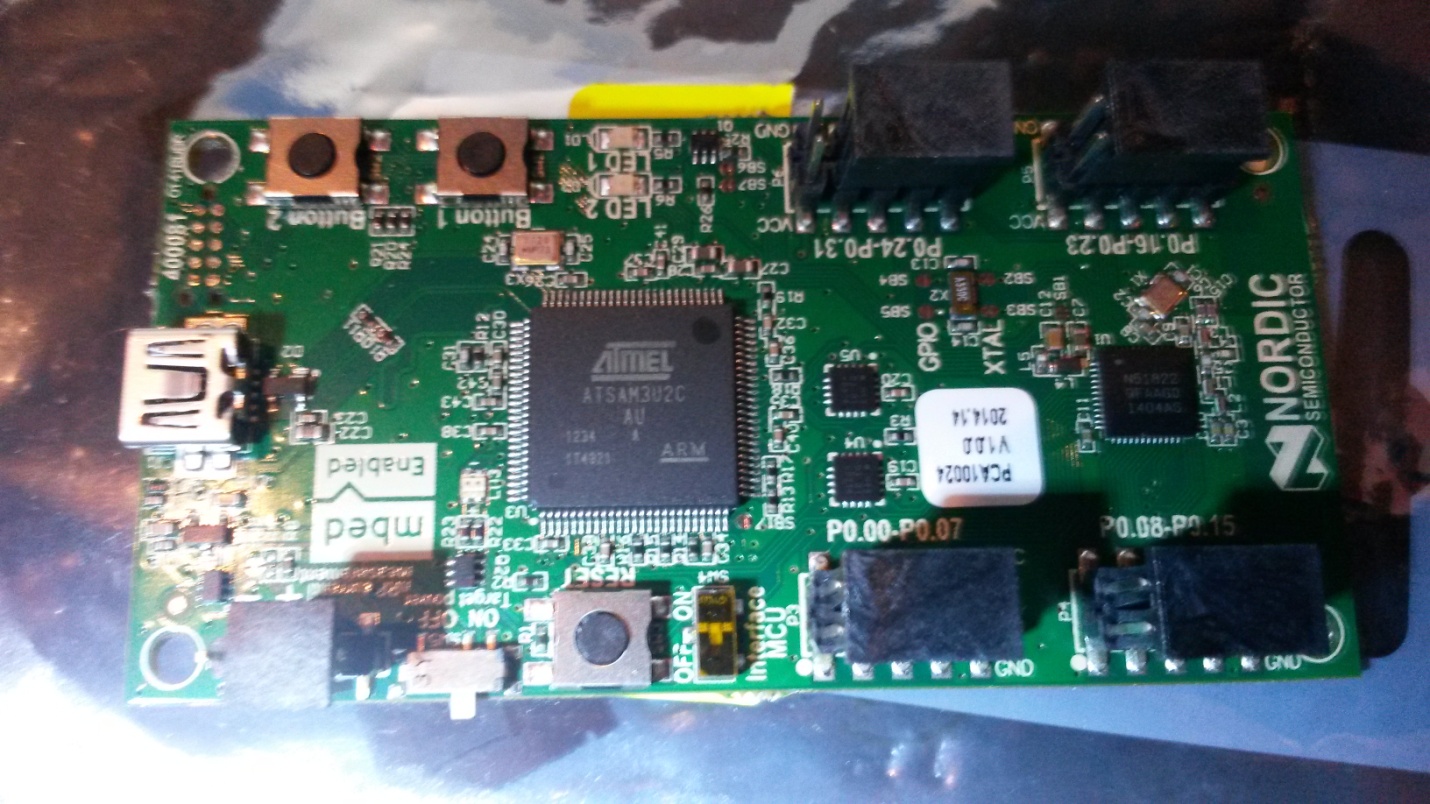
**UCSCX 4357, HW1 Hardware** Wojciech Wojciak

The mbed nRF51822-mKIT

<http://www.nordicsemi.com/eng/Products/Bluetooth-Smart-Bluetooth-low-energy/nRF51822-mKIT>

<http://www.nordicsemi.com/eng/nordic/download_resource/34975/4/43632221>



USB / UART

nRF51822 SoC

32768kHz crystal

ARM Cortex M0

Hardware description

The nRF51822 mbed kit board (PCA10024) is a standalone mbed enabled development board. The board is delivered with an unprogrammed nRF51822 chip.

The mbed kit board has the following key features:

* nRF51822 System on Chip combining Bluetooth v4.1-compliant 2.4 GHz multiprotocol radio
* ARM® Cortex®-M0 processor on a single chip optimized for ultra-low power operation.
* 31 GPIOs accessible on pin headers.
* USB drag and drop programming.
* USB virtual COM port for serial terminal

The mbed nRF51822-mKIT Evaluation kit is a stand-alone platform for rapid prototyping of *Bluetooth*® Smart designs with the nRF51822 SoC. The kit is compatible with the mbed *Bluetooth*Smart API which simplifies use of the *Bluetooth* low energy protocol stack.The nRF51822-mKIT gives access to all GPIO pins via pin headers and incorporates a coin-cell battery holder for portability enabling in-situ evaluation and test. Each board has 2 buttons, 2 LEDs, DC/DC converter circuit (optionally enabled in software), power supply and current measurement pins.The nRF51822-mKIT is compatible with a range of mbed software libraries based on the Cortex Microcontroller Software Interface Standard.

ARM Cortex M0

<http://www.arm.com/products/processors/cortex-m/cortex-m0.php>

**HW1 Software**

/\* UCSCX 4357 - HW1 – Wojciech Wojciak

write functions converting gray o binary and binary to gray

/\*

/\* exemplary (4bit) Gray / Binary table

Gray <-> Binary

0000 0000 0

0001 0001 1

0011 0010 2

0010 0011 3

0110 0100 4

0111 0101 5

0101 0110 6

0100 0111 7

1100 1000 8

1101 1001 9

1111 1010 A

1110 1011 B

1010 1100 C

1011 1101 D

1001 1110 E

1000 1111 F

\*/

#include <stdio.h>

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* FUNCTION bin2gray

INPUT: unsigned int

OUTPUT: unsigned int

ALGORITHM: return input shifted right once and XOR-ed with original input

\*/

unsigned int bin2gray(unsigned int bin)

{

return ( bin >> 1 ) ^ bin;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* FUNCTION gray2bin

INPUT: unsigned int

OUTPUT: unsigned int

ALGORITHM: 1. shift input right once and use it as a mask.

2. XOR input with mask

3. repeat 1 & 2 until mask is 0.

\*/

unsigned int gray2bin(unsigned int gray)

{

unsigned int mask;

unsigned int ret = gray;

for( mask = gray >> 1; mask != 0; mask = mask >> 1 ) {

ret = ret ^ mask;

}

return ret;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* FUNCTION printUsage()\*/

void printUsage()

{

printf("usage: convertBG <mode> [<input>]\n");

printf(" mode: 0 - bin to gray\n");

printf(" mode: 1 - gray to bin\n");

printf(" mode: 2 - test 0-15\n");

printf(" input: integer number\n");

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int main( int argc, char\* argv[] )

{

if(argc < 2) {

printUsage();

return -1;

}

int mode = atoi(argv[1]);

if( argc==3 && mode > 1 ) {

printUsage();

return -1;

}

if( argc==2 && mode != 2) {

printUsage();

return -1;

}

unsigned int inp;

unsigned int i;

if(mode==0) {

inp = (unsigned int) atoi(argv[2]);

printf("binary %u converted to gray is: %u\n", inp, bin2gray(inp));

} else if(mode==1) {

inp = (unsigned int) atoi(argv[2]);

printf("gray %u converted to binary is: %u\n", inp, gray2bin(inp));

} else if(mode==2) {

for(i=0; i<16; i++) {

printf("binary %u converted to gray is: %u\n", i, bin2gray(i));

}

for(i=0; i<16; i++) {

printf("gray %u converted to binary is: %u\n", i, gray2bin(i));

}

} else {

printUsage();

}

return 0;

}