Fontys University of Applied Sciences

Software & Technology - 6th Semester

Assignment: **GPIO**

Elviro Pereira Junior, **2719584**
Rafal Grasman, **289290**

17/04/2018

Research Results

Device Driver Type

First we looked into the different device drivers such as block and char device driver. The main difference between this two is that the char device performs actions immediately (unbuffered) and is used for sending/receiving single bytes of data and a block device is usually to send blocks (multiple bytes) of data which are buffered.[6]

To solve this assignment we decided that we would write a char device driver. The reason we chose a char driver instead of a block driver is because we'll be sending 1 byte at a time and not a whole block of data, and need the action to be performed immediately.

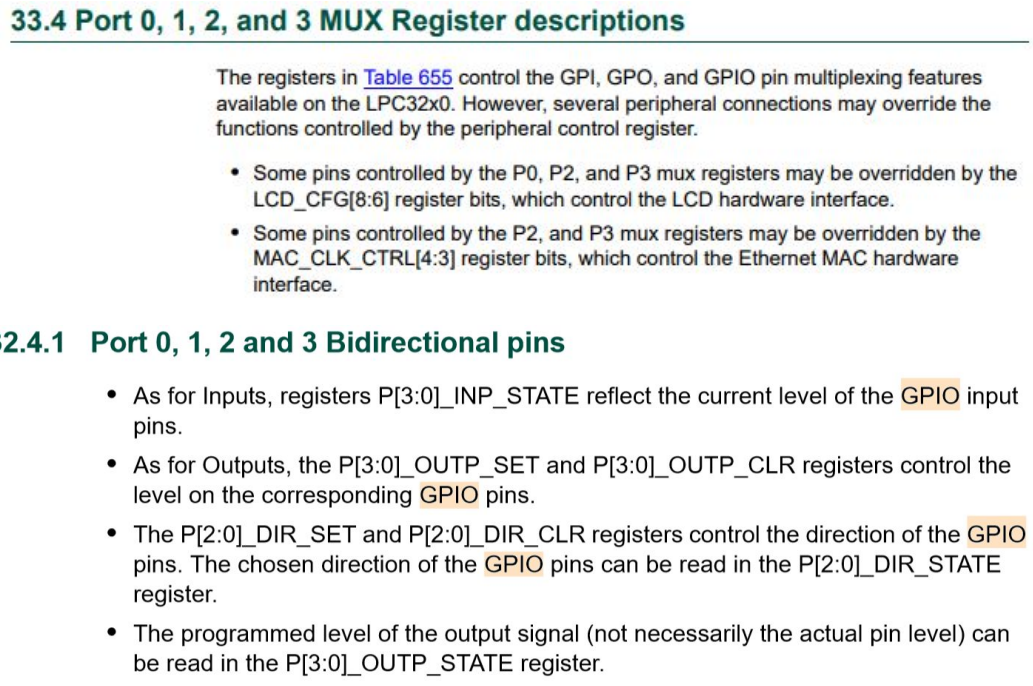
GPIO on LPC3250 board

The board has (from documentation) 38 GPIO pins, that can each be individually controlled by controlling 4 ports.

Each ports works in the same way[2]:

1) Set the MUX[3] to the correct values to enable GPIO pins (bitmasking)
2) Set the DIR to the correct values to make pins INPUT/OUTPUT (bitmasking)
3) The value can be read with OUTP_STATE and INP_STATE (depending on DIR) and written with OUTP_SET (when DIR = OUTPUT) (also bitmasking)

2



33.4 Port 0, 1, 2, and 3 MUX Register descriptions

The registers in Table 655 control the GPI, GPO, and GPIO pin multiplexing features available on the LPC32x0. However, several peripheral connections may override the functions controlled by the peripheral control register.

- Some pins controlled by the P0, P2, and P3 mux registers may be overridden by the LCD_CFG[8:6] register bits, which control the LCD hardware interface.
- Some pins controlled by the P2, and P3 mux registers may be overridden by the MAC_CLK_CTRL[4:3] register bits, which control the Ethernet MAC hardware interface.

32.4.1 Port 0, 1, 2 and 3 Bidirectional pins

- As for Inputs, registers P[3:0]_INP_STATE reflect the current level of the GPIO input pins.
- As for Outputs, the P[3:0]_OUTP_SET and P[3:0]_OUTP_CLR registers control the level on the corresponding GPIO pins.
- The P[2:0]_DIR_SET and P[2:0]_DIR_CLR registers control the direction of the GPIO pins. The chosen direction of the GPIO pins can be read in the P[2:0]_DIR_STATE register.
- The programmed level of the output signal (not necessarily the actual pin level) can be read in the P[3:0]_OUTP_STATE register.

Registers and Addresses

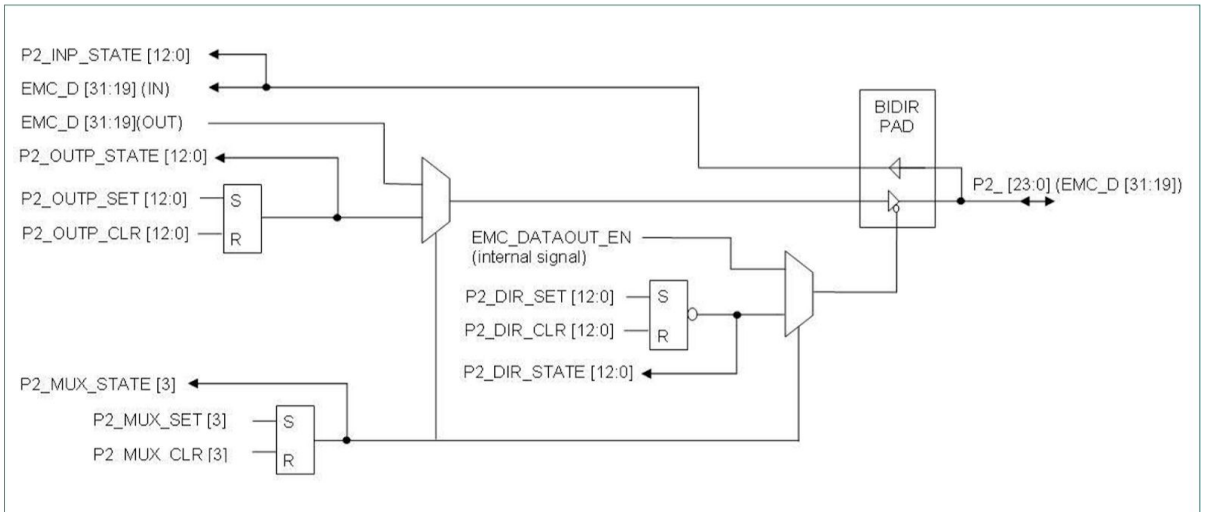
There are 10 registers per ports that are important. The registers work by only acknowledging ‘1’ bits and writing zeroes does nothing, which means there are usually 3 registers per function: A SET, a CLEAR and a STATE register per function.

In total, the following registers have been identified per port:

```
MUX_SET, MUX_CLR, MUX_STATE
OUTP_SET, OUTP_CLR, OUTP_STATE, INP_STATE
DIR_SET, DIR_CLR, DIR_STATE
```

Some registers are shared between ports, like port 2 and 3 MUX and DIR registers.

The port2 register for example has the following registers (logic overview):



[4] - Figure 125 from the datasheet

The addresses for the MUX registers are as follow [5]:

Table 655. Summary of GPIO Multiplexing registers

Address	Name	Description	Reset state	Access
Port 0				
0x4002 8120	P0_MUX_SET	Port 0 multiplexer set register. selects alternate functions on certain Port 0 pins.	-	WO
0x4002 8124	P0_MUX_CLR	Port 0 multiplexer clear register. selects default GPIO function on certain pins.	-	WO
0x4002 8128	P0_MUX_STATE	Port 0 multiplexer state register. Reads selection of alternate or I/O functions on Port 0 pins.	0x0000 0000	RO
Port 1				
0x4002 8130	P1_MUX_SET	Port 1 multiplexer set register. selects alternate functions on certain Port 1 pins.	-	WO
0x4002 8134	P1_MUX_CLR	Port 1 multiplexer clear register. selects default GPIO function on certain Port 1 pins.	-	WO
0x4002 8138	P1_MUX_STATE	Port 1 multiplexer state register. Reads selection of alternate or I/O functions on Port 1 pins.	0x0000 0000	RO
Port 2				
0x4002 8028	P2_MUX_SET	Port 2 multiplexer set register. selects alternate functions on certain Port 2 pins.	-	WO
0x4002 802C	P2_MUX_CLR	Port 2 multiplexer clear register. selects default GPIO functions on certain Port 2 pins.	-	WO
0x4002 8030	P2_MUX_STATE	Port 2 multiplexer state register. Reads selection of alternate or I/O functions on Port 2 pins.	0x0000 0000	RO
Port 3				
0x4002 8110	P3_MUX_SET	Port 3 multiplexer set register. Controls the selection of alternate functions on Port 3 pins.	-	WO
0x4002 8114	P3_MUX_CLR	Port 3 multiplexer clear register. selects default GPIO functions on certain Port 3 pins.	-	WO
0x4002 8118	P3_MUX_STATE	Port 3 multiplexer state register. Reads selection of alternate or I/O functions on Port 3 pins.	0x0000 0000	RO

The addresses for the mentioned 7 (with MUX it's 10) other registers per port are[7]:

Table 617. Summary of GPIO Data and Configuration registers

Address	Name	Description	Reset state	Access
Port 0				
0x4002 8040	P0_INP_STATE	Port 0 Input pin state register. Reads the state of input pins.	-	RO
0x4002 8044	P0_OUTP_SET	Port 0 Output pin set register. Allows setting output pin(s).	-	WO
0x4002 8048	P0_OUTP_CLR	Port 0 Output pin clear register. Allows clearing output pin(s).	-	WO
0x4002 804C	P0_OUTP_STATE	Port 0 Output pin state register. Reads the state of output pins.	-	RO
0x4002 8050	P0_DIR_SET	Port 0 direction set register. Configures P0 I/O pins as outputs.	-	WO
0x4002 8054	P0_DIR_CLR	Port 0 direction clear register. Configures P0 I/O pins as inputs.	-	WO
0x4002 8058	P0_DIR_STATE	Port 0 direction state register. Reads direction of P0 I/O pins.	0	RO
Port 1				
0x4002 8060	P1_INP_STATE	Port 1 Input pin state register. Reads the state of P1 pins.	-	RO
0x4002 8064	P1_OUTP_SET	Port 1 Output pin set register. Sets output value of P1 pins.	-	WO
0x4002 8068	P1_OUTP_CLR	Port 1 Output pin clear register. Clears output value of P1 pins.	-	WO
0x4002 806C	P1_OUTP_STATE	Port 1 Output pin state register. Reads state of P1 output pins.	-	RO
0x4002 8070	P1_DIR_SET	Port 1 direction set register. Configures Port 1 I/O pins as outputs.	-	WO
0x4002 8074	P1_DIR_CLR	Port 1 direction clear register. Configures Port 1 I/O pins as inputs.	-	WO
0x4002 8078	P1_DIR_STATE	Port 1 direction state register. Reads direction value Port 1 I/O pins.	0	RO

Table 617. Summary of GPIO Data and Configuration registers

Address	Name	Description	Reset state	Access
Port 2				
0x4002 801C	P2_INP_STATE	Port 2 Input pin state register. Reads the state of Port 2 GPIO pins.	-	RO
0x4002 8020	P2_OUTP_SET	Port 2 Output pin set register. Sets Port 2 GPIO output value.	-	WO
0x4002 8024	P2_OUTP_CLR	Port 2 Output pin clear register. Clears Port 2 GPIO output value.	-	WO
0x4002 8010	P2_DIR_SET	Port 2 and Port 3 GPIO direction set register. configures direction of I/O pins P2.[23:0] and GPIO_[5:0].	-	WO
0x4002 8014	P2_DIR_CLR	Port 2 and Port 3 GPIO direction clear register. configures direction of I/O pins P2.[23:0] and GPIO_[5:0].	-	WO
0x4002 8018	P2_DIR_STATE	Port 2 and Port 3 GPIO direction state register. Reads pin direction status for I/O pins P2.[23:0] and GPIO_[5:0].	0	RO
Port 3				
0x4002 8000	P3_INP_STATE	Port 3 Input pin state register. Reads the state of GPIO[5:0] and GPI input pins.	-	RO
0x4002 8004	P3_OUTP_SET	Port 3 output pin set register. Sets GPIO[5:0] output and GPO_[23:0] pin(s).	-	WO
0x4002 8008	P3_OUTP_CLR	Port 3 output pin clear register. Clears GPIO_[5:0] output and GPO_[23:0] pin(s).	-	WO
0x4002 800C	P3_OUTP_STATE	Port 3 output pin state register. Reads the state of GPIO_[5:0] output and GPO_[23:0] pin(s).	-	RO

For all pins, the documentation says: Direction OUTPUT = '1', Direction INPUT = '0', HIGH state = '1', LOW state = '0'. All PIN mappings overlap per port for each function (they are at the same bitmasks) which makes generalizing easier. As such only the pin STATE registers will be shown as they indicate at which bit which PIN is located.

Each register controls or reads different states of gpio pins (when the MUX is set correctly):
For port 0:

32.5.1 P0 Input Pin State register (P0_INP_STATE - 0x4002 8040)

The P0_INP_STATE register is a read-only register that provides the state of GPIO pins P0.7 through p0.0. This allows reading the pin values when the pins are used as GPIOs.

Table 618. P0 Input Pin State register for EMC address pins (P0_INP_STATE - 0x4002 8040)

P0_INP_STATE	Function	Description
0	P0.0 / I2S1RX_CLK	Reflects the state of the P0.0 pin.
1	P0.1 / I2S1RX_WS	Reflects the state of the P0.1 pin.
2	P0.2 / I2S0RX_SDA | (LCDVD[4])	Reflects the state of the P0.2 pin.
3	P0.3 / I2S0RX_CLK | (LCDVD[5])	Reflects the state of the P0.3 pin.
4	P0.4 / I2S0RX_WS | (LCDVD[6])	Reflects the state of the P0.4 pin.
5	P0.5 / I2S0TX_SDA | (LCDVD[7])	Reflects the state of the P0.5 pin.
6	P0.6 / I2S0TX_CLK | (LCDVD[12])	Reflects the state of the P0.6 pin.
7	P0.7 / I2S0TX_WS | (LCDVD[13])	Reflects the state of the P0.7 pin.

For port 1:

Table 625. P1 Input Pin State register for EMC address pins (P1_INP_STATE - 0x4002 8060)

P1_INP_STATE	Function	Description
0	EMC_A[0] / P1.0	Reflects the state of the P1.0 pin.
1	EMC_A[1] / P1.1	Reflects the state of the P1.1 pin.
2	EMC_A[2] / P1.2	Reflects the state of the P1.2 pin.
3	EMC_A[3] / P1.3	Reflects the state of the P1.3 pin.
4	EMC_A[4] / P1.4	Reflects the state of the P1.4 pin.
5	EMC_A[5] / P1.5	Reflects the state of the P1.5 pin.
6	EMC_A[6] / P1.6	Reflects the state of the P1.6 pin.
7	EMC_A[7] / P1.7	Reflects the state of the P1.7 pin.
8	EMC_A[8] / P1.8	Reflects the state of the P1.8 pin.
9	EMC_A[9] / P1.9	Reflects the state of the P1.9 pin.
10	EMC_A[10] / P1.10	Reflects the state of the P1.10 pin.
11	EMC_A[11] / P1.11	Reflects the state of the P1.11 pin.
12	EMC_A[12] / P1.12	Reflects the state of the P1.12 pin.
13	EMC_A[13] / P1.13	Reflects the state of the P1.13 pin.
14	EMC_A[14] / P1.14	Reflects the state of the P1.14 pin.
15	EMC_A[15] / P1.15	Reflects the state of the P1.15 pin.
16	EMC_A[16] / P1.16	Reflects the state of the P1.16 pin.
17	EMC_A[17] / P1.17	Reflects the state of the P1.17 pin.
18	EMC_A[18] / P1.18	Reflects the state of the P1.18 pin.
19	EMC_A[19] / P1.19	Reflects the state of the P1.19 pin.
20	EMC_A[20] / P1.20	Reflects the state of the P1.20 pin.
21	EMC_A[21] / P1.21	Reflects the state of the P1.21 pin.
22	EMC_A[22] / P1.22	Reflects the state of the P1.22 pin.
23	EMC_A[23] / P1.23	Reflects the state of the P1.23 pin

For port 2:

Table 632. Input Pin State register for EMC pins (P2_INP_STATE - 0x4002 801C)

P2_INP_STATE	Function	Description
0	EMC_D[19] / P2.0	Reflects the state of the P2.0 input pin.
1	EMC_D[20] / P2.1	Reflects the state of the P2.1 input pin.
2	EMC_D[21] / P2.2	Reflects the state of the P2.2 input pin.
3	EMC_D[22] / P2.3	Reflects the state of the P2.3 input pin.
4	EMC_D[23] / P2.4	Reflects the state of the P2.4 input pin.
5	EMC_D[24] / P2.5	Reflects the state of the P2.5 input pin.
6	EMC_D[25] / P2.6	Reflects the state of the P2.6 input pin.
7	EMC_D[26] / P2.7	Reflects the state of the P2.7 input pin.
8	EMC_D[27] / P2.8	Reflects the state of the P2.8 input pin.
9	EMC_D[28] / P2.9	Reflects the state of the P2.9 input pin.
10	EMC_D[29] / P2.10	Reflects the state of the P2.10 input pin.
11	EMC_D[30] / P2.11	Reflects the state of the P2.11 input pin.
12	EMC_D[31] / P2.12	Reflects the state of the P2.12 input pin.
31:13	Reserved	The value read from a reserved bit is not defined.

For port 3:

Table 638. Input Pin State Register (P3_INP_STATE - 0x4002 8000)

P3_INP_STATE	Function	Description
0	GPI_0 / I2S1RX_SDA	Reflects the general purpose input pin GPI_0.
1	GPI_1 / SERVICE_N	Reflects the general purpose input pin GPI_1 / SERVICE_N.
2	GPI_2 / CAP2.0 | (ENET_RXD3)	Reflects the general purpose input pin GPI_2.
3	GPI_3	Reflects the general purpose input pin GPI_3.
4	GPI_4 / SPI1_BUSY	Reflects the general purpose input pin GPI_4 / SPI1_BUSY.
5	GPI_5 / U3_DCD	Reflects the general purpose input pin GPI_5.
6	GPI_6 / HSTIM_CAP | (ENET_RXD2)	Reflects the general purpose input pin GPI_6 / HSTIM_CAP.
7	GPI_7	Reflects the general purpose input pin GPI_7.
8	GPI_8 / KEY_COL6 / SPI2_BUSY | (ENET_RX_DV)	Reflects the general purpose input pin GPI_8 / KEY_COL6 / SPI2_BUSY.
9	GPI_9 / KEY_COL7 | (ENET_COL)	Reflects the general purpose input pin GPI_9 / KEY_COL7
10	GPIO_0	Reflects the general purpose I/O pin GPIO_0.
11	GPIO_1	Reflects the general purpose I/O pin GPIO_1.
12	GPIO_2 / KEY_ROW6 | (ENET_MDC)	Reflects the general purpose I/O pin GPIO_2 / KEY_ROW6
13	GPIO_3 / KEY_ROW7 | (ENET_MDIO)	Reflects the general purpose I/O pin GPIO_3 / KEY_ROW7
14	GPIO_4 / SSEL1 | (LCDVD[22])	Reflects the general purpose I/O pin GPIO_4.
15	GPI_15 / U1_RX | (CAP1.0)	Reflects the state of the input pin GPI_15 / U1_RX.
16	GPI_16 / U2_HCTS | (U3_CTS)	Reflects the state of the input pin GPI_16 / U2_HCTS.
17	GPI_17 / U2_RX | (U3_DSR)	Reflects the state of the input pin GPI_17 / U2_RX.
18	GPI_18 / U3_RX	Reflects the state of the input pin GPI_18 / U3_RX.
19	GPI_19 / U4_RX	Reflects the general purpose input pin GPI_19 / U4_RX.
20	GPI_20 / U5_RX	Reflects the state of the input pin GPI_20 / U5_RX.
21	GPI_21 / U6_IRRX	Reflects the state of the input pin GPI_21 / U6_IRRX.
22	GPI_22 / U7_HCTS / CAP0.1 | (LCDCLKIN)	Reflects the state of the input pin GPI_22 / U7_HCTS.

PORT2 Does seem to have a OUTP_STATE (fig 125 suggests that) register, but it is not explicitly mentioned anywhere. We found that at offset +4 (educated guess based on relation of OUTP_STATE on other ports with OUTP_CLR) PORT2 does have OUTP_STATE (from OUTP_CLR) like all other ports, which works as expected.

Port Mappings

The customer wants to be able to access GPIO by specifying a connector (J1, J2, J3) and a PIN (1 ... 64), this mean that each pin has to be traced from the OEM board [8] to the development board [9] with the connectors, this are the results (other GPIO are not available through PINS, and PORT1 is not connected to the dev board therefore we were unable to TEST it):

Port 0:

P0.0 - X1-106 - P1.8 - J3.40
P0.1 - X1-107 - P1.9 - J2.24
P0.2 - X1-31 - P2.6 - J2.11
P0.3 - X1-32 - P2.7 - J2.12
P0.4 - X1-33 - P2.8 - J2.13
P0.5 - X1-34 - P2.9 - J2.14
P0.6 - X1-90 - P1.22 - J3.33
P0.7 - X1-91 - P1.23 - J1.27

Port 2:

P2.0 - X1-120 - P2.22 - J3.47
P2.1 - X1-121 - P2.23 - J3.56
P2.2 - X1-122 - P2.25 - J3.48
P2.3 - X1-123 - P2.26 - J3.55
P2.4 - X1-124 - P2.27 - J3.49
P2.5 - X1-125 - P2.30 - J3.58
P2.6 - X1-126 - P2.31 - J3.50
P2.7 - X1-116 - P2.14 - J3.45

P2.8 - X1-176 - P3.27 - J1.49
P2.9 - X1-178 - P3.26 - J1.50
P2.10 - X1-180 - P3.25 - J1.51
P2.11 - X1-182 - P3.24 - J1.52
P2.12 - X1-184 - P3.23 - J1.53

Port 3:

GPIO_00 - X1-117 - P2.15 - J3.54
GPIO_01 - X1-118 - P2.19 - J3.46

GPIO_04 - X1-96 - P1.28 - J3.36
GPIO_05 - X1-85 - P1.13 - J1.24

Design Decisions

To make accessing pins easier it has been chosen to map the connectors and pins on the board to the GPIO, so the following layout is being suggested by the kernel module upon load (when debugging):

```
# ls /dev
es6_gpio.J2.13-value      es6_gpio.J3.49-value       es6_gpio.J2.13-direction
es6_gpio.J3.49-direction
es6_gpio.J1.24-direction  es6_gpio.J2.14-direction  es6_gpio.J3.50-direction
es6_gpio.J1.24-value      es6_gpio.J2.14-value      es6_gpio.J3.50-value
es6_gpio.J1.27-direction  es6_gpio.J2.24-direction  es6_gpio.J3.54-direction
es6_gpio.J1.27-value      es6_gpio.J2.24-value      es6_gpio.J3.54-value
es6_gpio.J1.49-direction  es6_gpio.J3.33-direction  es6_gpio.J3.55-direction
es6_gpio.J1.49-value      es6_gpio.J3.33-value      es6_gpio.J3.55-value
es6_gpio.J1.50-direction  es6_gpio.J3.36-direction  es6_gpio.J3.56-direction
es6_gpio.J1.50-value      es6_gpio.J3.36-value      es6_gpio.J3.56-value
es6_gpio.J1.51-direction  es6_gpio.J3.40-direction  es6_gpio.J3.58-direction
es6_gpio.J1.51-value      es6_gpio.J3.40-value      es6_gpio.J3.58-value
es6_gpio.J1.52-direction  es6_gpio.J3.45-direction
es6_gpio.J1.52-value      es6_gpio.J3.45-value
es6_gpio.J1.53-direction  es6_gpio.J3.46-direction
es6_gpio.J1.53-value      es6_gpio.J3.46-value
es6_gpio.J2.11-direction  es6_gpio.J3.47-direction
es6_gpio.J2.11-value      es6_gpio.J3.47-value
es6_gpio.J2.12-direction  es6_gpio.J3.48-direction
es6_gpio.J2.12-value      es6_gpio.J3.48-value
```

Writing a ‘1’ to direction will make the pin an OUTPUT, writing a ‘0’ and INPUT. Direction can also be read and the current direction will be output. Reading value will output the current state (1 being high, 0 being low) of the pin. A ‘1’ can be written to the value to make the pin high, and a ‘0’ to make the pin low (when in output mode).

Concurrency

- Global variable

To keep the code concurrent, in this assignment we make use of global variables for read only, in cases such as determine register address or figuring out a minor number of a device and map it to the right pin. (pins are board specific)

- Non Global variable

We made use of non global, or use of private data in situations where the operation/function in case is used by more than one process at the time in order to avoid race conditions.

Readability

To hence the readability we decided to keep repeated data such connectors names, port address, functions, device information, etc, into structs to keep the code short and avoid unnecessary repetition.

```
struct ConnectionMapping
{
      Port <- which connector on the board this mapping corresponds to
      Pin <- which pin on the connector this mapping corresponds to
};

struct PortAddresses
{
      ENABLE_MASK <- Which mask to use to enable GPIO
      MUX_SET <- address of the register to enable GPIO with mask
      MUX_CLR <- “” to disable GPIO with mask
      MUX_STATE <- “” to determine current state
      INP_STATE <- “” to determine GPIO input values
      OUTP_SET <- “” to set GPIO pins to high
      OUTP_CLR <- “” to set GPIO pins to low
      OUTP_STATE <- “” to determine high/low state of a GPIO pin
      DIR_SET <- “” to set the GPIO pins direction to output
      DIR_CLR <- “” to set the GPIO pins direction to input
      DIR_STATE <- “” to determine if a GPIO pin is out-/input
      MAPPING_OFFSET <- at which bit offset the mapping starts in all
registers
      MAPPING_PINS <- how many pins are mapped
      struct ConnectionMapping MAPPING[] <- this is do determine which
index (bit offset = index + mapping offset) the connector/pin
combination corresponds to for this port in the registers
};
```

Mux_SET & Mux_CLR address swap

Per generalized code design, the ENABLE_MASK is set in MUX's to enable GPIO, and set to 0 to disable GPIO on a specific port. Because enabling the GPIO pins for ports 0 and 3 happen by setting the mask to 1's in the CLR registers, the CLR and SET registers have been swapped in the port information structure to allow for consistent code usage without implementing special cases for ports 0 and 3.

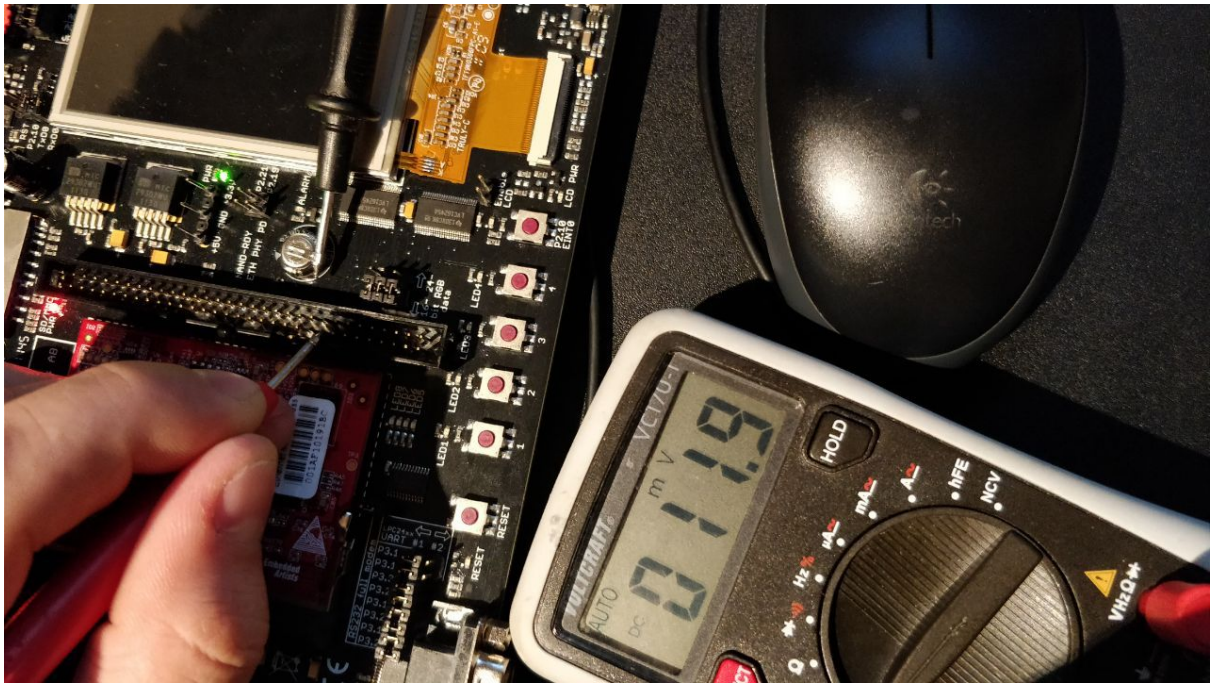
Testing

All different ports have been tested by “echo”-ing and “cat”-ing to and from the mentioned devices in the design. Each direction has been tested (input and output) and each state (high low), and each combination of these two. For each Port (0, 2 and 3) one pin was tested. Port 1 does not expose any external GPIO connectors, so it cannot be tested by wiring into the development board.(Note: board-specific)

ScreenShots

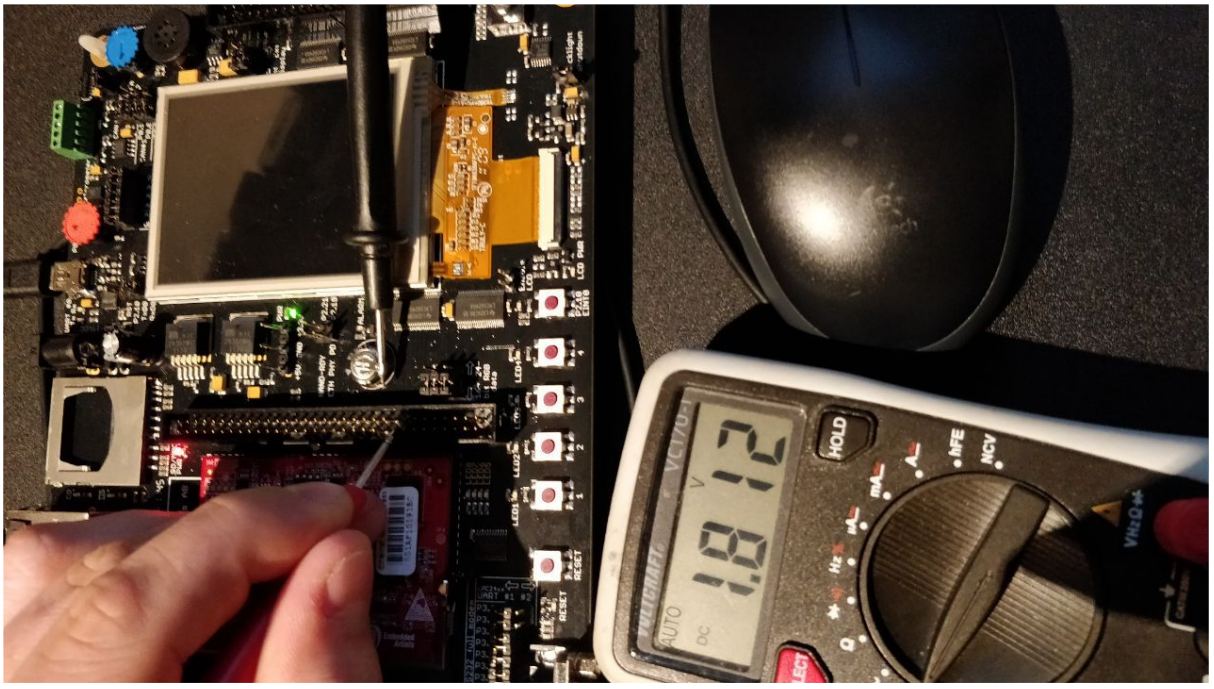
(with cleaned out debug output)
Test for both output and input (both HIGH and LOW), have been done for J3.33, J3.47, J3.58 and J3.54 (to mix different ports and bit positions in ports). In the images and console example output J3.47 is show, photos and console output is similar to J3.47 for all other tested pins / ports.

```
# echo "1" > /dev/es6_gpio.J3.47-direction
# cat /dev/es6_gpio.J3.47-direction
1
# cat /dev/es6_gpio.J3.47-value
0
```



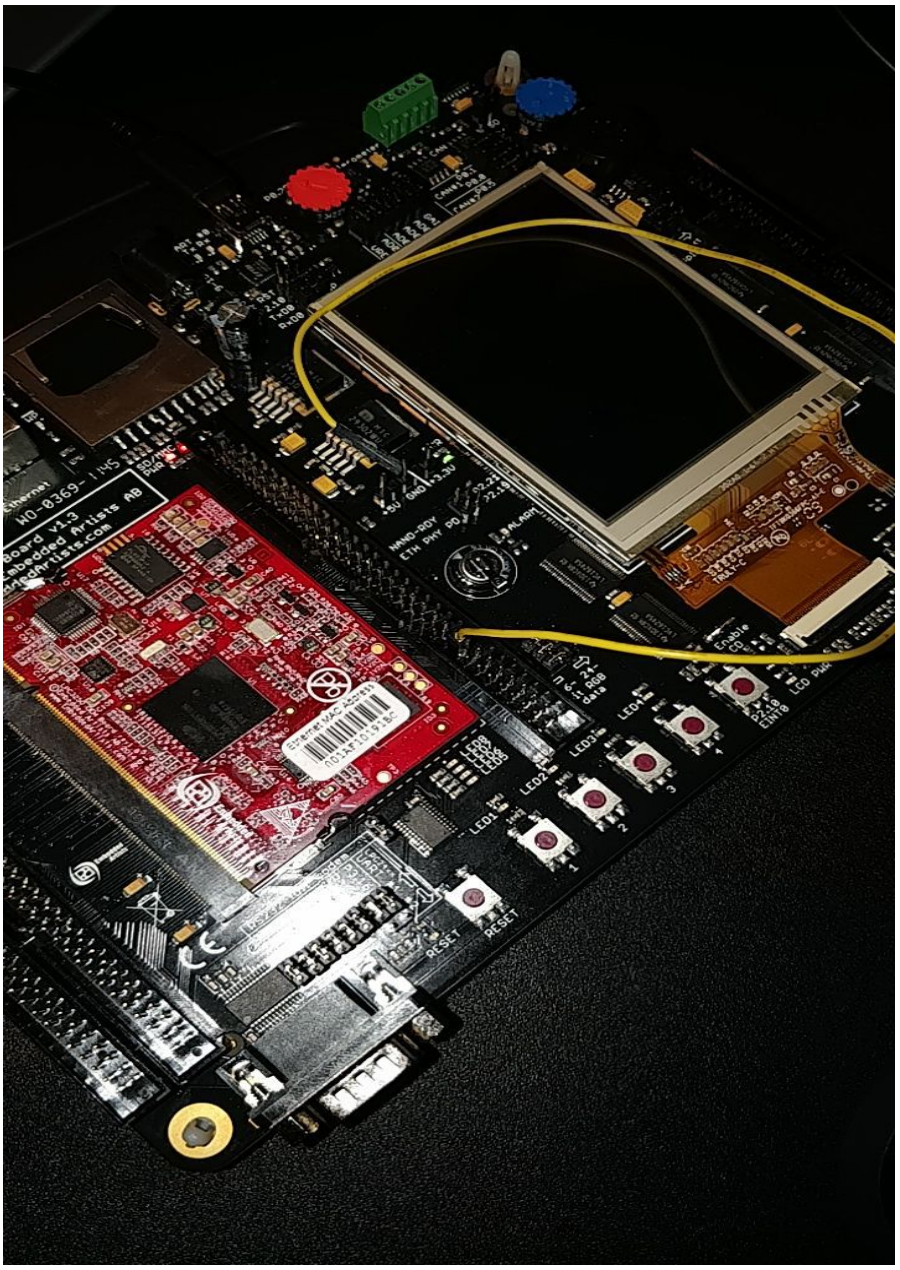
J3.47 (OUTPUT) turned to LOW

```
# echo "1" > /dev/es6_gpio.J3.47-value
# cat /dev/es6_gpio.J3.47-value
1
```



J3.47 (OUTPUT) turned to HIGH

```
# echo "0" > /dev/es6_gpio.J3.47-value
# cat /dev/es6_gpio.J3.47-value
0
```



J3.47 (INPUT) forced to LOW by connecting to GND

```
# echo "0" > /dev/es6_gpio.J3.47-direction
# cat /dev/es6_gpio.J3.47-direction
0
# # here floating
# cat /dev/es6_gpio.J3.47-value
1
# # here connecting to ground
# cat /dev/es6_gpio.J3.47-value
0
```

Reference

1 - DataSheet-UM10326, page 22
2 - DataSheet-UM10326, page 604
3 - DataSheet-UM10326, page 605
4 - DataSheet-UM10326, page 607
5 - DataSheet-UM10326, page 631
6 - mknod manual page, http://man7.org/linux/man-pages/man2/mknod.2.html
7 - DataSheet-UM10326, page 609
8 - LPC32x0_OEM_Board_v1.3.pdf
9 - QVGA_Base_Board_v1.2.pdf