ECE 544 Nexys4IO Driver Documentation

Roy Kravitz

Version 1.0
26-Dec-14

Include Files

```
#include "xil_types.h"
#include "xstatus.h"
#include "stdbool.h"
#include "Nexys4IO_l.h"
```

Macros

Bit Masks*Bit masks for the Nexys4IO registers.*

All of the registers in the Nexys4IO periheral are 32-bits wide

- #define **NEXYS4IO_BTNR_MASK** 0x00010000
- #define **NEXYS4IO_BTNL_MASK** 0x00020000
- #define **NEXYS4IO_BTND_MASK** 0x00040000
- #define **NEXYS4IO_BTNU_MASK** 0x00080000
- #define **NEXYS4IO_BTNC_MASK** 0x00100000
- #define **NEXYS4IO_ALLBTNS_MASK** 0x001F0000
- #define **NEXYS4IO_ALLSWITCHES_MASK** 0x0000FFFF
- #define **NEXYS4IO_LEDS_MASK** 0x0000FFFF
- #define **NEXYS4IO_RGB_BLUEDC_MASK** 0x000000FF
- #define **NEXYS4IO_RGB_GREENDC_MASK** 0x0000FF00
- #define **NEXYS4IO_RGB_REDDC_MASK** 0x00FF0000
- #define **NEXYS4IO_RGB_CHEN_MASK** 0x00000007
- #define **NEXYS4IO_SSEG_DIG0_MASK** 0x0000001F
- #define **NEXYS4IO_SSEG_DIG1_MASK** 0x000007C0
- #define **NEXYS4IO_SSEG_DIG2_MASK** 0x0001F000
- #define **NEXYS4IO_SSEG_DIG3_MASK** 0x007C0000
- #define **NEXYS4IO_SSEG_DECPTS_MASK** 0x0F000000
- #define **NEXYS4IO_SSEG_DECPT3_MASK** 0x08000000
- #define **NEXYS4IO_SSEG_DECPT2_MASK** 0x04000000
- #define **NEXYS4IO_SSEG_DECPT1_MASK** 0x02000000
- #define **NEXYS4IO_SSEG_DECPT0_MASK** 0x01000000

Literals and constants

Literals and constants used for selecting specific devicees

- enum **_NX4IO_btns** { **BTNR**, **BTNL**, **BTND**, **BTNU**, **BTNC** }
- enum **_NX4IO_rgbleds** { **RGB1** = 1, **RGB2** = 2 }
- enum **_NX4IO_ssegbanks** { **SSEGLO** = 1, **SSEGHI** = 2 }
- enum **_NX4IO_ssegdigits** { **DIGIT0**, **DIGIT1**, **DIGIT2**, **DIGIT3**, **DIGIT4**, **DIGIT5**, **DIGIT6**, **DIGIT7** }
- enum **_NX4IO_charcodes** { **CC_0**, **CC_1**, **CC_2**, **CC_3**, **CC_4**, **CC_5**, **CC_6**, **CC_7**, **CC_8**, **CC_9**, **CC_A**, **CC_B**, **CC_C**, **CC_D**, **CC_E**, **CC_F**, **CC_SEGa**, **CC_SEGb**, **CC_SEGc**, **CC_SEGd**, **CC_SEGe**, **CC_SEGf**, **CC_SEGg**, **CC_SPACE**, **CC_UCH**, **CC_UCL**, **CC_UCR**, **CC_LCL**, **CC_LCR**, **CC_LCY**, **CC_BLANK**, **CC_BLANK1** }
- enum **_NX410_decpts** { **DP_0** = 0x0, **DP_1** = 0x01, **DP_2** = 0x04, **DP_3** = 0x8, **DP_ALL** = 0xF, **DP_NONE** = 0x0 }
- int **NX4IO_initialize** (u32 BaseAddr)
- u32 **NX4IO_getBTNSW_IN** (void)
- u8 **NX4IO_getBtns** (void)
- u16 **NX4IO_getSwitches** (void)
- bool **NX4IO_ isPressed** (enum _NX4IO_btns)
- u32 **NX4IO_getLEDS_DATA** (void)
- void **NX4IO_setLEDs** (u32 ledvalue)

- u32 **NX4IO_RGBLED_getRGB_DATA** (enum _NX4IO_rgbleds led)
- u32 **NX4IO_RGBLED_getRGB_CNTRL** (enum _NX4IO_rgbleds led)
- void **NX4IO_RGBLED_setRGB_DATA** (enum _NX4IO_rgbleds led, u32 data)
- void **NX4IO_RGBLED_setRGB_CNTRL** (enum _NX4IO_rgbleds led, u32 cntrl)
- void **NX4IO_RGBLED_setDutyCycle** (enum _NX4IO_rgbleds led, u8 redDC, u8 greenDC, u8 blueDC)
- void **NX4IO_RGBLED_setChnlEn** (enum _NX4IO_rgbleds led, bool en_red, bool en_green, bool en_blue)
- u32 **NX4IO_SSEG_getSSEG_DATA** (enum _NX4IO_ssegbanks bank)
- void **NX4IO_SSEG_setSSEG_DATA** (enum _NX4IO_ssegbanks bank, u32 data)
- int **NX4IO_SSEG_setDigit** (enum _NX4IO_ssegbanks bank, enum _NX4IO_ssegdigits digit, enum _NX4IO_charcodes cc)
- int **NX4IO_SSEG_setDecPt** (enum _NX4IO_ssegbanks bank, enum _NX4IO_ssegdigits digit, bool on)
- int **NX410_SSEG_setAllDigits** (enum _NX4IO_ssegbanks bank, u8 dig3, u8 dig2, u8 dig1, u8 dig0, u8 dp)
- int **NX4IO_SSEG_putU16Hex** (enum _NX4IO_ssegbanks bank, u16 data)
- int **NX4IO_SSEG_putU32Hex** (u32 data)
- int **NX4IO_SSEG_putU32Dec** (u32 data, bool trim)

Detailed Description

Author:

Roy Kravitz (`roy.kravitz@pdx.edu`)

Copyright:

Portland State University, 2014, 2015

This header file contains identifiers and driver functions for the Nexys4IO custom peripheral. The peripheral provides access to the Nexys4 pushbuttons and slide switches, the LEDs, the RGB LEDs, and the Seven Segment display on the Digilent Nexys4 board.

```
MODIFICATION HISTORY:
```

Ver	Who	Date	Changes
1.00a	rhk	12/20/14	First release of driver

Function Documentation

bool NX4IO_isPressed (enum _NX4IO_btns *btnslct*)

returns the state of the selected pushbutton

Reads the pushbuttons and checks if the selected button is pressed (i.e. 1)

Parameters:

btnslct	selects which button to check

Returns:

true if the button is pressed, false otherwise

Note:

No error checking is done on btnslct. default returns false

int NX410_SSEG_setAllDigits (enum _NX4IO_ssegbanks *bank*, u8 *dig3*, u8 *dig2*, u8 *dig1*, u8 *dig0*, u8 *dp*)

sets all of the digits and the decimal points in the selected bank of digits

Writes a new value to all of the digit in the SSEG_DATA for the selected bank. Also writes the decimal points. It is expected that the digits be in the set specified by enum _NX4IO_ssegdigits but no checking is done. Instead each digit will be written with the lower 5 bits of the 8-bit digit value(s) passed into the function.

The Nexys4 board has two 4-digit seven segment display banks. SSEGLO includes digits 3-0 (rightmost digits). SSEGHI includes digits 7-4 (leftmost digits)

Parameters:

bank	is used to select which of the SSEG_DATA data registers to write
dig3,dig2,dig1	and dig0 are the new digit values. dig3 is the leftmost digit in the bank, dig0 is the rightmost digit in the bank.
dp	is the enw value for the decimal points. Only the least significant four bits are used with bit[3] being the decimal point to the right of dig3 and so on to bit[0] and dig0.

Returns:

XST_SUCCESS if the operation succeeds. XST_FAILURE if the operation failed (i.e. one of the parameters was invalid)

Note:

No checking is done on the bank select. Doesn't write invalid register

u8 NX4IO_getBtns (void)

returns the current value of the pushbuttons

Reads BTNSW_IN, masks the buttons and right justifies them

Parameters:

None	

Returns:

current value of the pushbuttons right justified in an 8-bit field

Note:

Buttons are returned as follows: 0 0 0 BTNC BTNU BTND BTNL BTNR

u32 NX4IO_getBTNSW_IN (void)

returns the current value BTNSW_IN.

Returns the raw value of BTNSW_IN. No formatting or bit masking is done

Parameters:

None	

Returns:

current value of the pushbutons and switches. No error checking is done and the bit formatting is as shown in the datasheet.

Note:

See the NEXYS4IO Datasheet for the format of the BTNSW_IN register

u32 NX4IO_getLEDS_DATA (void)

returns the current value LEDS_DATA.

Returns the raw value of LEDS_DATA. No formatting or bit masking is done

Parameters:

None	

Returns:

current value of the LEDs register. No error checking is done and the bit formatting is as shown in the datasheet.

Note:

See the NEXYS4IO Datasheet for the format of the LEDS_DATA register

u16 NX4IO_getSwitches (void)

returns the current value of the slide switches

Reads BTNSW_IN, masks the switches and right justifies them

Parameters:

None	

Returns:

current value of the switches right justified in a 16-bit field

Note:

switches are returned as follows: SW15..SW0

int NX4IO_initialize (u32 *BaseAddr*)

Initialize the NEXYS4IO peripheral driver

Saves the Base address of the NEXYS4IO peripheral and runs the selftest

Parameters:

BaseAddr	is the base address of the NEXYS4IO register set

Returns:

- XST_SUCCESS Initialization was successful.

Note:

This function can hang if the peripheral was not created correctly
The Base Address of the NEXYS4IO peripheral will be in xparameters.h

u32 NX4IO_RGBLED_getRGB_CNTRL (enum _NX4IO_rgbleds *led*)

returns the RGB_CNTRL register for the selected RGB LED

Reads and returns the raw value of the selected RGB LED control register

Parameters:

led	is used to select which of the RGB LED control registers to read

Returns:

Raw (not formatted) value of the selected RGB LED data register

Note:

See the NEXYS4IO Datasheet for the format of the RGB_CNTRL register
No checking is done on the RGB LED select. Returns 0 as default

u32 NX4IO_RGBLED_getRGB_DATA (enum _NX4IO_rgbleds *led*)

returns the RGB_DATA register for the selected RGB LED

Reads and returns the raw value of the selected RGB LED data register

Parameters:

led	is used to select which of the RGB LED data registers to read

Returns:

Raw (not formatted) value of the selected RGB LED data register

Note:

See the NEXYS4IO Datasheet for the format of the RGB_DATA register
No checking is done on the RGB LED select. Returns 0 as default

void NX4IO_RGBLED_setChnlEn (enum _NX4IO_rgbleds *led*, bool *en_red*, bool *en_green*, bool *en_blue*)

sets the enables for the Red, Green and Blue channels of the selected RBG LED

Formats the channel enables bits per the RGB_CNTRL register specification and writes the new duty cycles to the RGB_CNTRL register. A channel is enabled by writing a 1 to its channel enable bit.

Parameters:

led	is used to select which of the RGB LED data registers to read
en_red	is the enable bit for the red LED in the RGB LED
en_green	is the enable bit for the green LED in the RGB LED
en_blue	is the enable bit for the blue LED in the RGB LED

Returns:

NONE

Note:

See the NEXYS4IO Datasheet for the format of the RGB_CNTRL register
No checking is done on the RGB LED select. Doesn't write invalid register

void NX4IO_RGBLED_setDutyCycle (enum _NX4IO_rgbleds *led*, u8 *redDC*, u8 *greenDC*, u8 *blueDC*)

sets the duty cycle of the Red, Green and Blue channels of the selected RBG LED

Formats the PWM duty cycles per the RGB_DATA register specification and writes the new duty cycles to the RGB_DATA register. Duty cycles should be expressed as an 8-bit unsigned number.

Momentarily disables the R, G, and B channels then changes the values and re-enables the channels that were previously enabled.

Parameters:

led	is used to select which of the RGB LED data registers to read
redDC	us the new duty cycle for the red LED in the RGB LED
greenDC	us the new duty cycle for the green LED in the RGB LED
blueDC	us the new duty cycle for the blue LED in the RGB LED

Returns:

NONE

Note:

See the NEXYS4IO Datasheet for the format of the RGB_DATA register
No checking is done on the RGB LED select. Doesn't write invalid register
The RGB PWM logic in Nexys4IO limits the duty cycle to 50% as recommended in the Digilent Nexys4 User guide

void NX4IO_RGBLED_setRGB_CNTRL (enum _NX4IO_rgbleds *led*, u32 *cntrl*)

sets the RGB_CNTRL register for the selected RGB LED

Writes a new value to the selected RGB LED channel enable register

Parameters:

led	is used to select which of the RGB LED data registers to write
cntrl	is the value to be written to the register

Returns:

NONE

Note:

See the NEXYS4IO Datasheet for the format of the RGB_CNTRL register
No checking is done on the RGB LED select. Doesn't write invalid register

void NX4IO_RGBLED_setRGB_DATA (enum _NX4IO_rgbleds *led*, u32 *data*)

sets the RGB_DATA register for the selected RGB LED

Writes a new value to the selected RGB LED data register

Parameters:

led	is used to select which of the RGB LED data registers to write
data	is the value to be written to the register

Returns:

NONE

Note:

See the NEXYS4IO Datasheet for the format of the RGB_DATA register
No checking is done on the RGB LED select. Doesn't write invalid register

void NX4IO_setLEDs (u32 *ledvalue*)

sets the LEDS_DATA register

Lights (or not) the LEDS.

Parameters:

ledvalue	is the value to write to the LEDS_DATA register. The unused bits are masked out and set to 0

Returns:

NONE

Note:

See the NEXYS4IO Datasheet for the format of the LEDS_DATA register

u32 NX4IO_SSEG_getSSEG_DATA (enum _NX4IO_ssegbanks *bank*)

returns the SSEG_DATA register for the selected bank of digits

Reads and returns the raw value of the selected SSEG_DATA data register. The Nexys4 board has two 4-digit seven segment display banks. SSEGLO includes digits 3-0 (rightmost digits). SSEGHI includes digits 7-4 (leftmost digits)

Parameters:

bank	is used to select which display bank register to read

Returns:

Raw (not formatted) value of the selected data register

Note:

See the NEXYS4IO Datasheet for the format of the SSEG_DATA register
No checking is done on the bank select. Returns 0 as default

int NX4IO_SSEG_putU16Hex (enum _NX4IO_ssegbanks *bank*, u16 *data*)

writes a 16-bit unsigned hex number to the selected display bank

Breaks a 16-bit binary number (u16) into individual digits and displays them on the selected seven segment display bank.

The Nexys4 board has two 4-digit seven segment display banks. SSEGLO includes digits 3-0 (rightmost digits). SSEGHI includes digits 7-4 (leftmost digits)

Parameters:

bank	is used to select which of the SSEG_DATA data registers to write
data	is the 16-bit unsigned number that will be displayed in hex

Returns:

XST_SUCCESS if the number was displayed correctly. XST_FAILURE if the operation failed (i.e. one of the parameters was invalid)

Note:

See the NEXYS4IO Datasheet for the character code table and the format of the SSEG_DATA registers No checking is done on the bank select. Doesn't write invalid registers.

int NX4IO_SSEG_putU32Dec (u32 *data*, bool *trim*)

writes a 32-bit unsigned decimal number to the selected display bank

Breaks a 32-bit binary number (u32) into individual digits and displays them on all 8 digits of the segment display. Converts the number to packed BCD so that it can be displayed. Does bounds checking on the maximum number that can be displayed (0 to 99.999.999) and fails if the number is out of range. Trims leading 0's

The Nexys4 board has two 4-digit seven segment display banks. SSEGLO includes digits 3-0 (rightmost digits). SSEGHI includes digits 7-4 (leftmost digits)

Parameters:

data	is the 32-bit unsigned number that will be displayed in decimal
trim	is a boolean. If true, leading 0's ared converted to blanks

Returns:

XST_SUCCESS if the number was displayed correctly. XST_FAILURE if the operation failed (i.e. one of the parameters was invalid)

Note:

See the NEXYS4IO Datasheet for the character code table and the format of the SSEG_DATA registers

int NX4IO_SSEG_putU32Hex (u32 *data*)

writes a 32-bit unsigned hex number to the selected display bank

Breaks a 32-bit binary number (u32) into individual digits and displays them on all 8 digits of the segment display

The Nexys4 board has two 4-digit seven segment display banks. SSEGLO includes digits 3-0 (rightmost digits). SSEGHI includes digits 7-4 (leftmost digits)

Parameters:

data	is the 32-bit unsigned number that will be displayed in hex

Returns:

XST_SUCCESS if the number was displayed correctly. XST_FAILURE if the operation failed (i.e. one of the parameters was invalid)

Note:

See the NEXYS4IO Datasheet for the character code table and the format of the SSEG_DATA registers

int NX4IO_SSEG_setDecPt (enum _NX4IO_ssegbanks *bank*, enum _NX4IO_ssegdigits *digit*, bool *on*)

sets a single decimal point in the selected bank of digits

Changes the decimal point in the specified digit in the SSEG_DATA register for the selected bank. The boolean 'on' defines whether the decimal point is lit or not. If 'on' is true the decimal point is lit. If false, it is not. The digits and the other decimal point values are unchanged. Use **NX4IO_SSEG_setSSEG_DATA()** if you want to change more than one digit, or digit(s) and decimal points in a single operation

The Nexys4 board has two 4-digit seven segment display banks. SSEGLO includes digits 3-0 (rightmost digits). SSEGHI includes digits 7-4 (leftmost digits)

Parameters:

bank	is used to select which of the SSEG_DATA data registers to write
digit	specifies which digit (7-4 or 3-0) will be changed
on	is the new state of the selected decimal point.

Returns:

XST_SUCCESS if the decimal point was changed. XST_FAILURE if the operation failed (i.e. one of the parameters was invalid)

Note:

See the NEXYS4IO Datasheet for the character code table and the format of the SSEG_DATA registers
No checking is done on the bank select. Doesn't write invalid register

int NX4IO_SSEG_setDigit (enum _NX4IO_ssegbanks *bank*, enum _NX4IO_ssegdigits *digit*, enum _NX4IO_charcodes *cc*)

sets a single digit in the selected bank of digits

Writes a new character code to the specified digit in the SSEG_DATA for the selected bank. The character code to write is checked to make sure it is in the range of the entries in the _NX4IO_charcodes table. The remaining digits (those not specified) and the decimal points are not modified. Use **NX4IO_SSEG_setSSEG_DATA()** if you want to change more than one digit, or digit(s) and decimal points in a single operation

The Nexys4 board has two 4-digit seven segment display banks. SSEGLO includes digits 3-0 (rightmost digits). SSEGHI includes digits 7-4 (leftmost digits)

Parameters:

bank	is used to select which of the SSEG_DATA data registers to write
digit	specifies which digit (7-4 or 3-0) will be changed
cc	is the new character code for the digit

Returns:

XST_SUCCESS if the digit was changed. XST_FAILURE if the operation failed (i.e. one of the parameters was invalid)

Note:

See the NEXYS4IO Datasheet for the character code table and the format of the SSEG_DATA registers
No checking is done on the bank select. The cc is checked to see if it is range and only written if it is in range. Doesn't write invalid register

void NX4IO_SSEG_setSSEG_DATA (enum _NX4IO_ssegbanks *bank*, u32 *data*)

sets the SSEG_DATA register for the selected bank of digits

Writes a new value to the selected SSEG_DATA data register. The Nexys4 board has two 4-digit seven segment display banks. SSEGLO includes digits 3-0 (rightmost digits). SSEGHI includes digits 7-4 (leftmost digits)

Parameters:

bank	is used to select which of the SSEG_DATA data registers to write
data	is the value to be written to the register

Returns:

NONE

Note:

See the NEXYS4IO Datasheet for the format of the SSEG_DATA register
No checking is done on the bank select. Doesn't write invalid register

Nexys4IO_l.h File Reference

```
#include "xil_types.h"
#include "xil_io.h"
#include "xstatus.h"
```

Macros

- #define **NEXYS4IO_mWriteReg**(BaseAddress, RegOffset, Data)   Xil_Out32((BaseAddress) + (RegOffset), (u32)(Data))
- #define **NEXYS4IO_mReadReg**(BaseAddress, RegOffset)   Xil_In32((BaseAddress) + (RegOffset))

  Registers*Register offsets for this device.*
  - #define **NEXYS4IO_BTNSW_IN_OFFSET**   0
  - #define **NEXYS4IO_LEDS_DATA_OFFSET**   4
  - #define **NEXYS4IO_RGB1_DATA_OFFSET**   8
  - #define **NEXYS4IO_RGB1_CNTRL_OFFSET**   12
  - #define **NEXYS4IO_RGB2_DATA_OFFSET**   16
  - #define **NEXYS4IO_RGB2_CNTRL_OFFSET**   20
  - #define **NEXYS4IO_SSEGLO_DATA_OFFSET**   24
  - #define **NEXYS4IO_SSEGHI_DATA_OFFSET**   28
  - #define **NEXYS4IO_RSVD00_OFFSET**   32
  - #define **NEXYS4IO_RSVD01_OFFSET**   36
  - #define **NEXYS4IO_RSVD02_OFFSET**   40
  - #define **NEXYS4IO_RSVD03_OFFSET**   44
  - #define **NEXYS4IO_RSVD04_OFFSET**   48
  - #define **NEXYS4IO_RSVD05_OFFSET**   52
  - #define **NEXYS4IO_RSVD06_OFFSET**   56
  - #define **NEXYS4IO_RSVD07_OFFSET**   60

Functions

- XStatus **NEXYS4IO_Reg_SelfTest** (u32 baseaddr)

Detailed Description

Author:

Roy Kravitz (`roy.kravitz@pdx.edu`)

Copyright:

Portland State University, 2014, 2015

This header file contains identifiers and low level driver functions for the Nexys4IO custom peripheral. The peripheral provides access to the Nexys4 pushbuttons and slide switches, the LEDs, the RGB LEDs, and the Seven Segment display on the Digilent Nexys4 board.

```
MODIFICATION HISTORY:
```

Ver	Who	Date	Changes
1.00a	rhk	12/20/14	First release of driver

Macro Definition Documentation

#define NEXYS4IO_mReadReg(BaseAddress, RegOffset) Xil_In32((BaseAddress) + (RegOffset))

Read a value from a NEXYS4IO register. A 32 bit read is performed. If the component is implemented in a smaller width, only the least significant data is read from the register. The most significant data will be read as 0.

Parameters:

BaseAddress	is the base address of the NEXYS4IO device.
RegOffset	is the register offset from the base to write to.

Returns:

Data is the data from the register.

Note:

C-style signature: u32 **NEXYS4IO_mReadReg(u32 BaseAddress, unsigned RegOffset)**

#define NEXYS4IO_mWriteReg(BaseAddress, RegOffset, Data) Xil_Out32((BaseAddress) + (RegOffset), (u32)(Data))

Write a value to a NEXYS4IO register. A 32 bit write is performed. If the component is implemented in a smaller width, only the least significant data is written.

Parameters:

BaseAddress	is the base address of the NEXYS4IOdevice.
RegOffset	is the register offset from the base to write to.
Data	is the data written to the register.

Returns:

None

Note:

C-style signature: void **NEXYS4IO_mWriteReg(u32 BaseAddress, unsigned RegOffset, u32 Data)**

Function Documentation

XStatus NEXYS4IO_Reg_SelfTest (u32 *baseaddr*)

Run a self-test on the Nexys4IO driver/device.

If the hardware system is not built correctly, this function may never return to the caller.

Parameters:

baseaddr_p	is the base address of the NEXYS4IO instance to be worked on.

Returns:

- XST_SUCCESS if all self-test code passed
- XST_FAILURE if any self-test code failed

Note:

Caching must be turned off for this function to work.
Self test may fail if data memory and device are not on the same bus.
This test assume the existence of a Serial port in the system (used for xil_printf)

Run a self-test on the driver/device. Note this may be a destructive test if resets of the device are performed.

If the hardware system is not built correctly, this function may never return to the caller.

Parameters:

baseaddr_p	is the base address of the NEXYS4IOinstance to be worked on.

Returns:

- XST_SUCCESS if all self-test code passed
- XST_FAILURE if any self-test code failed

Note:

Caching must be turned off for this function to work.
Self test may fail if data memory and device are not on the same bus.
Assume the existence of a serial port for xil_printf()

Index