NIOS Character SW

Last updated 10/12/20

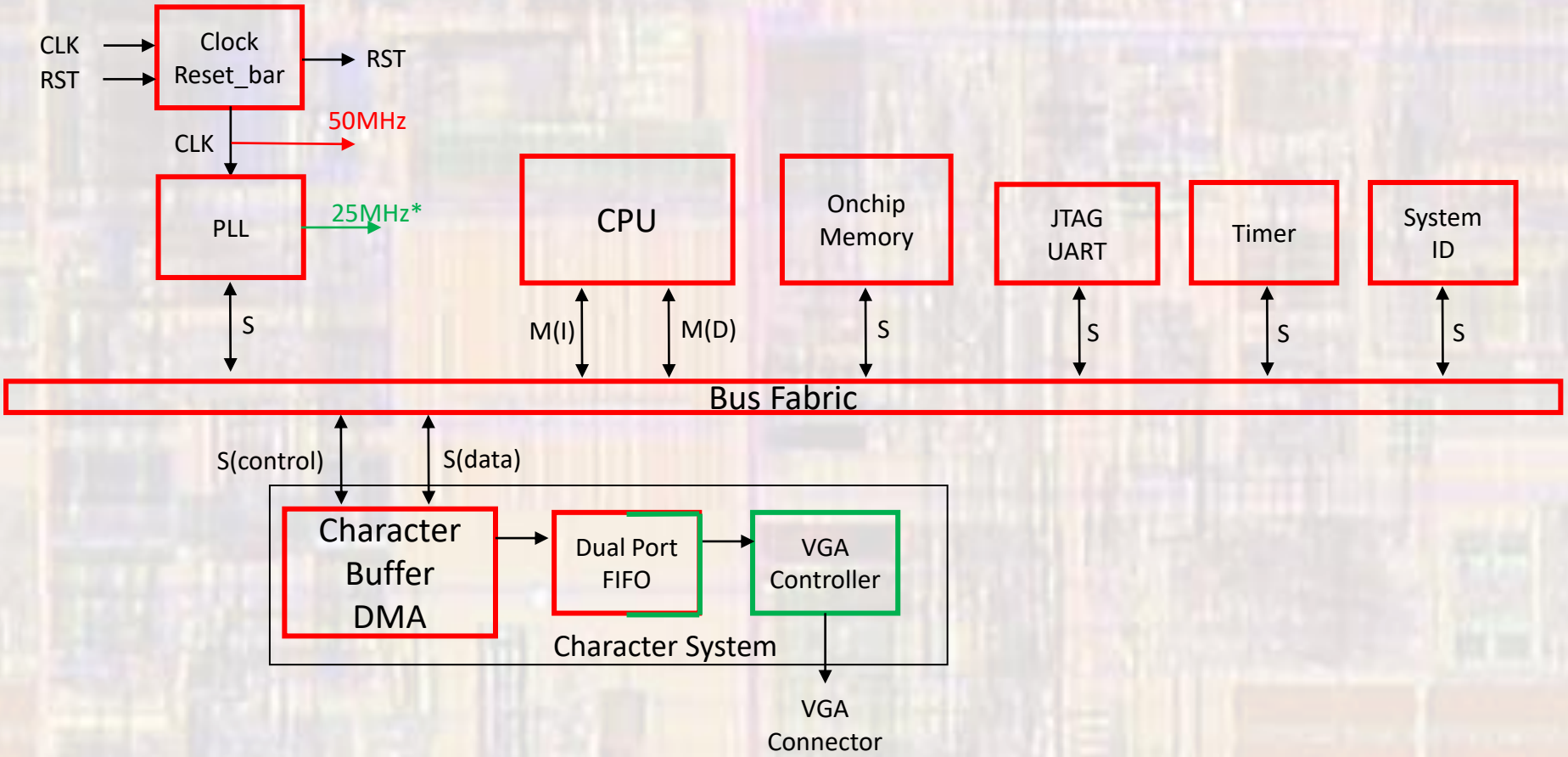
NIOS II Character Display - SW

These slides describe the development of software for a moderately complex NIOS Processor using the Character Buffer IP

Upon completion: You should be able implement your own software on a NIOS processor using the Character Buffer IP

NIOS II Character Display - SW

- Character Buffer Block Diagram



25MHz** - VGA clk

NIOS II Character Display - SW

- Create Eclipse System
  - Open NIOSII software
    - Tools → NIOSII Software Build Tools for Eclipse
  - Create the BSP
    - File → New → NIOSII Application and BSP from template
    - Blank Template
  - Edit the BSP
    - Right click on the BSP, NIOS II → BSP Editor
    - Change the properties for small systems
      - Small C library
      - Reduced device drivers
  - Re-Generate the BSP

NIOS II Character Display - SW

- Create Eclipse System
  - In the BSP – under drivers/inc
    - Open altera_up_avalon_video_character_buffer_with_dma.h
    - Find the video character buffer structure name

```
17  typedef struct alt_up_char_buffer_dev {
18      /// @brief character mode device structure
19      /// @sa Developing Device Drivers for the HAL in Nios II Software Developer's Handbook
20      alt_dev dev;
21      /// @brief the control register's slave base address
22      unsigned int ctrl_reg_base;
23      /// @brief the character buffer's slave base address
24      unsigned int buffer_base;
25      /// @brief the character resolution in x direction
```

    - Create a pointer of this type

```
// define a pointer of type char buffer type
// to use as a reference in the dma functions
//
alt_up_char_buffer_dev * char_buf_dev;
```

NIOS II Character Display - SW

- Create Eclipse System
  - In the BSP – under drivers/inc
    - Open altera_up_avalon_video_character_buffer_with_dma.h
    - Find the function to open the character buffer dma device

```
50 ////////////////////////////////////////////////////////////////////////////////
51 // direct operation functions
52 /**
53  * @brief Opens the character buffer device specified by <em> name </em>
54  *
55  * @param name -- the character buffer component name in SOPC Builder.
56  *
57  * @return The corresponding device structure, or NULL if the device is not found
58  **/
59 alt_up_char_buffer_dev* alt_up_char_buffer_open_dev(const char* name);
60
```

NIOS II Character Display - SW

- Create Eclipse System
  - In the BSP – in system.h
    - Open the device and assign it to the previously defined pointer

```
 * video_character_buffer_with_dma_0_avalon_char_buffer_slave configuration
 *
 */

#define ALT_MODULE_CLASS_video_character_buffer_with_dma_0_avalon_char_buffer_slave altera_up_avalon_video_character_buffer_with_dma
#define VIDEO_CHARACTER_BUFFER_WITH_DMA_0_AVALON_CHAR_BUFFER_SLAVE_BASE 0x0
#define VIDEO_CHARACTER_BUFFER_WITH_DMA_0_AVALON_CHAR_BUFFER_SLAVE_IRQ -1
#define VIDEO_CHARACTER_BUFFER_WITH_DMA_0_AVALON_CHAR_BUFFER_SLAVE_IRQ_INTERRUPT_CONTROLLER_ID -1
#define VIDEO_CHARACTER_BUFFER_WITH_DMA_0_AVALON_CHAR_BUFFER_SLAVE_NAME "/dev/video_character_buffer_with_dma_0_avalon_char_buffer_slave"
#define VIDEO_CHARACTER_BUFFER_WITH_DMA_0_AVALON_CHAR_BUFFER_SLAVE_SPAN 8192
#define VIDEO_CHARACTER_BUFFER_WITH_DMA_0_AVALON_CHAR_BUFFER_SLAVE_TYPE "altera_up_avalon_video_character_buffer_with_dma"


/*
 * video_character_buffer_with_dma_0_avalon_char_control_slave configuration
 *
 */

#define ALT_MODULE_CLASS_video_character_buffer_with_dma_0_avalon_char_control_slave altera_up_avalon_video_character_buffer_with_dma
#define VIDEO_CHARACTER_BUFFER_WITH_DMA_0_AVALON_CHAR_CONTROL_SLAVE_BASE 0x5030
#define VIDEO_CHARACTER_BUFFER_WITH_DMA_0_AVALON_CHAR_CONTROL_SLAVE_IRQ -1
#define VIDEO_CHARACTER_BUFFER_WITH_DMA_0_AVALON_CHAR_CONTROL_SLAVE_IRQ_INTERRUPT_CONTROLLER_ID -1
#define VIDEO_CHARACTER_BUFFER_WITH_DMA_0_AVALON_CHAR_CONTROL_SLAVE_NAME "/dev/video_character_buffer_with_dma_0_avalon_char_control_slave"
#define VIDEO_CHARACTER_BUFFER_WITH_DMA_0_AVALON_CHAR_CONTROL_SLAVE_SPAN 8
#define VIDEO_CHARACTER_BUFFER_WITH_DMA_0_AVALON_CHAR_CONTROL_SLAVE_TYPE "altera_up_avalon_video_character_buffer_with_dma"

#endif /* __SYSTEM_H_ */
```

```
// open the Character Buffer control port
// name reference is in system.h
// "/dev/video_character_buffer_with_dma_0_avalon_char_buffer_slave"
//
char_buf_dev = alt_up_char_buffer_open_dev("/dev/video_character_buffer_with_dma_0");
```

NIOS II Character Display - SW

- Create Eclipse System
  - In the BSP – under drivers/inc
    - Open altera_up_avalon_video_character_buffer_with_dma.h
    - The remainder of the character buffer dma commands are in this file

This actually writes to the buffer

```
/**
 * @brief Draw a character at the location specified by <em>(x, y)</em> on the
 * VGA monitor with white color and transparent background
 *
 * @param ch -- the character to draw
 * @param x-- the \em x coordinate
 * @param y-- the \em y coordinate
 *
 * @return 0 for success, -1 for error (such as out of bounds)
 **/
int alt_up_char_buffer_draw(alt_up_char_buffer_dev *char_buffer, unsigned char ch,
unsigned int x, unsigned int y);
```

0 - 79

0 - 59

NIOS II Character Display - SW

- Create Eclipse System

This actually writes to the buffer

```
/**
 * @brief Draw a NULL-terminated text string at the location specified by <em>(x, y)</em>
 *
 * @param ch -- the character to draw
 * @param x-- the \em x coordinate
 * @param y-- the \em y coordinate
 *
 * @return 0 for success, -1 for error (such as out of bounds)
 **/
int alt_up_char_buffer_string(alt_up_char_buffer_dev *char_buffer, const char *ptr,
unsigned int x, unsigned int y);

/**
 * @brief Clears the character buffer's memory
 *
 * @return 0 for success
 **/
int alt_up_char_buffer_clear(alt_up_char_buffer_dev *char_buffer);
```

Strings do not wrap around the display

NIOS II Character Display - SW

- Create Eclipse System
  - Write a program to print some characters to the screen

```
//////////////////
// Include files
//////////////////
#include "altera_up_avalon_video_character_buffer_with_dma.h"
#include "system.h"
#include <stdio.h>
```

NIOS II Character Display - SW

- Create Eclipse System
  - Write a program to print some characters to the screen

```
int main(void){
// define a pointer of type char buffer type
// to use as a reference in the dma functions
//
alt_up_char_buffer_dev * char_buf_dev;

// open the Character Buffer port
// name reference is in system.h
// "/dev/video_character_buffer_with_dma_0_avalon_char_buffer_slave"
//
char_buf_dev =
alt_up_char_buffer_open_dev("/dev/video_character_buffer_with_dma_0");

// Check for error and output to the console
//
if ( char_buf_dev == NULL)
   printf ("Error: could not open character buffer device \n");
else
   printf ("Opened character buffer device \n");
```

NIOS II Character Display - SW

- Create Eclipse System
  - Write a program to print some characters to the screen

```
    // Print some text to the screen
    //
    char text = 'X';
    char text_top_row[40] = "Altera DE10_lite\0";
    char text_bottom_row[40] = "Character Buffer\0";

    /* output text message near the middle of the VGA monitor */
    alt_up_char_buffer_clear(char_buf_dev);
    alt_up_char_buffer_draw(char_buf_dev, text, 0, 0);
    alt_up_char_buffer_draw(char_buf_dev, text, 0, 59);
    alt_up_char_buffer_draw(char_buf_dev, text, 79, 0);
    alt_up_char_buffer_draw(char_buf_dev, text, 79, 59);
    alt_up_char_buffer_string(char_buf_dev, text_top_row, 20,20);
    alt_up_char_buffer_string(char_buf_dev, text_bottom_row, 40,40);

    // end program message
    printf ("Program complete \n");

  return 0;
}
```

NIOS II Character Display - SW

- Create Eclipse System
  - Compile the software
    - Select the code file (char.c)
    - Project → Build Project
    - Right Click on the project → run as → Nios II Hardware

NIOS II Character Display - SW

- Create Eclipse System