Embedded Systems with uCLinux based on FPGA

by

Siddharth Kamath

February 17th, 2010
An Embedded System based on FPGA

• The Hardware architecture
• Design and selection of hardware platform
• uCLinux porting of hardware
• Application: Designing an mpeg2-player
FPGA’s so far…

- Configured by the customer
- Low NRE as compared to an ASIC
- Supports high speed computing
- Compatible with most modern I/O’s
- Processors can be implemented in soft-cores
Hardware architecture

- SOPC (System on a Programmable Chip) used to design the hardware
- SOPC – A software by Altera which automates connecting software – hardware components
- Peripherals designed:
  - Ethernet controller
  - SD card controller
  - LCD controller
  - USB controller

Fig. 1 Block diagram of hardware architecture
Hardware Platform

- Development board: Altera DE2-70
- Core: Cyclone II EP2C70 FPGA
  - 68416 logic elements
  - 250 M4K RAM blocks (1152000 bits)
- Other on-board peripherals:
  - FLASH and SDRAM
  - Ethernet
  - USB
  - Audio card
  - RS232
  - PS/2

Fig. 2 Hardware platform of embedded system
Hardware design using SOPC builder

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
<th>sys_clk</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>cpu</td>
<td>Nios II Processor</td>
<td></td>
<td>0x00000000</td>
</tr>
<tr>
<td>flash_tristate_bridge</td>
<td>Avalon-MM Tristate Bridge</td>
<td>sys_clk</td>
<td></td>
</tr>
<tr>
<td>ext_flash</td>
<td>Flash Memory (QPI)</td>
<td>sys_clk</td>
<td>0x00000000</td>
</tr>
<tr>
<td>sdram</td>
<td>SDRAM Controller</td>
<td>sys_clk</td>
<td>0x02000000</td>
</tr>
<tr>
<td>sys_clk_timer</td>
<td>Interval Timer</td>
<td>sys_clk</td>
<td>0x08001800</td>
</tr>
<tr>
<td>ISP1362</td>
<td>ISP 1362 Module if</td>
<td>multiple</td>
<td></td>
</tr>
<tr>
<td>lcd_i2c_scl</td>
<td>PIO (Parallel)</td>
<td>sys_clk</td>
<td>0x08001800</td>
</tr>
<tr>
<td>lcd_i2c_en</td>
<td>PIO (Parallel)</td>
<td>sys_clk</td>
<td>0x08001900</td>
</tr>
<tr>
<td>lcd_i2c_sdat</td>
<td>PIO (Parallel)</td>
<td>sys_clk</td>
<td>0x08001920</td>
</tr>
<tr>
<td>ps2_0</td>
<td>PS2 Serial Port</td>
<td>sys_clk</td>
<td>0x08001000</td>
</tr>
<tr>
<td>dm9000</td>
<td>DM9000A Module</td>
<td></td>
<td>0x00000000</td>
</tr>
<tr>
<td>uart</td>
<td>UART (RS-232 Serial Port)</td>
<td>sys_clk</td>
<td>0x08001880</td>
</tr>
<tr>
<td>lcd_sgdma</td>
<td>Scatter-Gather DMA Controller</td>
<td>sys_clk</td>
<td>0x08001400</td>
</tr>
<tr>
<td>lcd_ta_sgdma_to_fifo</td>
<td>Avalon-ST Timing Adapter</td>
<td>sys_clk</td>
<td></td>
</tr>
<tr>
<td>lcd_pixel_fifo</td>
<td>On-Chip FIFO Memory</td>
<td>sys_clk</td>
<td></td>
</tr>
<tr>
<td>lcd_ta_fifo_to_dma</td>
<td>Avalon-ST Timing Adapter</td>
<td>sys_clk</td>
<td></td>
</tr>
<tr>
<td>lcd_64_to_32_bite_dma</td>
<td>Avalon-ST Data Format Adapter</td>
<td>sys_clk</td>
<td></td>
</tr>
<tr>
<td>lcd_pixel_converter</td>
<td>Pixel Converter (BGR0 -&gt; BGR)</td>
<td>sys_clk</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 3 Design of hardware of FPGA
Software Design with uCLinux

- uClinux incorporated as an OS on the NIOS II processor
- uClinux originally designed for Motorola 68000 chip
- Why uClinux?
  - uClinux is designed for processors without MMU
  - Open source
  - Powerful network functions
  - Efficient and robust file system
- Kernel version 2.6.26 used included in nios2-linux-20080619.tar
- Custom ports can be added
The uClinux architecture

- All devices listed under the OS can be used
- Example: SD card as the root device, USB as keyboard, PS/2 as the mouse
Applications

- Nano-X: An open source GUI for embedded systems
- SDL: The Simple Direct Media Layer which is a cross platform multimedia library
- Links: A fast text and graphical web browser
- BusyBox: A set of many common UNIX utilities as a single executable

In short…..Its an on-board computer system on FPGA
Design of an MPEG2-player

- Design based on libmpeg2- a stream decoder
- Audio device needed to play MPEG2 files
- The on-board audio device on DE2-70 used
- Challenge: No audio drivers under the NIOS II platform
Control of the audio device

• Method 1: Writing device drivers
  • Disadvantage: Time consuming
    High rate of error
• Method 2: Directly read/write data using the port address
  • Advantage: Hassle free
    Easy to debug
Initializing the audio device

- Only write access required in an audio device
- Initiate audio device by initializing the I2C port
- Set the sample rate and audio channel
- Write data to audio device
- Control FIFO of audio components
Code excerpt to initialize the I2C port

```c
int AUDIO_Init(void)
{
    int complete = 1;
    printf("\n\n"));
    if (complete)
        complete = I2CWrite(15, 0x0000);
    if (complete)
        complete = I2CWrite(9, 0x0000);
    if (complete)
        complete = I2CWrite(0, 0x0017);
    if (complete)
        complete = I2CWrite(1, 0x0017);
    if (complete)
        complete = I2CWrite(2, 0x005E);
    if (complete)
        complete = I2CWrite(3, 0x005E);
    if (complete)
        complete = I2CWrite(4, 0x0015 | 0x20 | 0x08 | 0x01);
    if (complete)
        complete = I2CWrite(5, 0x0000);
    if (complete)
        complete = I2CWrite(6, 0);
    if (complete)
        complete = I2CWrite(7, 0x0042);
    if (complete)
        complete = I2CWrite(8, 0x0002);
    if (complete)
        complete = I2CWrite(9, 0x0001);
    printf("audio_Init %s\n", complete ? "success" : "fail");
    return complete;
}
```
MPEG2-player design flowchart

1. Start
2. Open file in read/write mode (mpeg_open)
3. Read the header
4. Sample rate for audio (mpeg_has_audio)
5. Frame rate for video (mpeg_has_video)
6. Decode files and send to their respective audio or video devices
7. Close the mpeg2 file
Performance and Conclusion

- System is highly reconfigurable
- Limited processor performance
- Soft-core processors have more flexibility
- High scalability – Peripherals can be easily added
- Multicore softcore processors can be designed to improve performance
References


