Embedded Image Capture

Alex Glass
Albert Jimenez
Nektarios Georgios Tsoutsos
## Control Signals

<table>
<thead>
<tr>
<th>Keyboard Button</th>
<th>Functionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spacebar</td>
<td>Freeze image</td>
</tr>
<tr>
<td>Up and down keys</td>
<td>Change screen mode</td>
</tr>
<tr>
<td>Delete</td>
<td>Request image deletion</td>
</tr>
<tr>
<td>Left key</td>
<td>Move to next image</td>
</tr>
<tr>
<td>Right key</td>
<td>Move to previous image</td>
</tr>
<tr>
<td>Enter</td>
<td>Save image</td>
</tr>
<tr>
<td>Right Shift Enter</td>
<td>Send image to remote board</td>
</tr>
<tr>
<td>Left Shift Enter</td>
<td>Send compressed image to computer</td>
</tr>
</tbody>
</table>
# Sending Images

<table>
<thead>
<tr>
<th>Address Number</th>
<th>Functionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Send x position</td>
</tr>
<tr>
<td>1</td>
<td>Send y position</td>
</tr>
<tr>
<td>2</td>
<td>Send request</td>
</tr>
<tr>
<td>3</td>
<td>Receive pixel</td>
</tr>
</tbody>
</table>

**Diagram: TV to VGA and Nios**

1. `pixel_req`
2. `x, y`
3. `pixel`
Image Data Structures

typedef unsigned char saved_images[MAX_NUM_IMAGES][3][IMG_X_SIZE][IMG_Y_SIZE];
// data structure for linking images
struct image
{
    struct image *prev;
    int is_taken
    int image_num;
    struct image *next;
};
UDP Server

Idle

start_of_data

Received File Size

file size

Received Data

data

Write File
UDP Client

- Start
- Send start of data packet
- Send file size packet
- Send files in 1 kb chunks
- Done
UDP Apps

```
alexander@vaio: $ cd cvs/csee4840/alex/udptest
alexander@vaio:udptest$ udpserver -v output
Udpserver listening for connections on port 9930
Received size packet 10.8.8.1 (file=575 bytes, packet=4 bytes)
Received data packet 1 from 10.8.8.1 (575 bytes)
Received 575 bytes, output written to file 'file1.jpg'
Received size packet 10.8.8.1 (file=5284 bytes, packet=4 bytes)
Received data packet 1 from 10.8.8.1 (1824 bytes)
Received data packet 2 from 10.8.8.1 (1824 bytes)
Received data packet 3 from 10.8.8.1 (1824 bytes)
Received data packet 4 from 10.8.8.1 (1824 bytes)
Received data packet 5 from 10.8.8.1 (1824 bytes)
Received data packet 6 from 10.8.8.1 (84 bytes)
Received 5284 bytes, output written to file 'file2.jpg'

alexander@vaio:udptest$ udpclient -v test.jpg
alexander@vaio:udptest$ udpclient -v 10.8.8.1 lenna.jpg
Udpclient sending to 10.8.8.1:9930
Sending packet 1 (1824 bytes)
Sending packet 2 (1824 bytes)
Sending packet 3 (1824 bytes)
Sending packet 4 (1824 bytes)
Sending packet 5 (1824 bytes)
Sending packet 6 (84 bytes)
Transmitted 5284 bytes of file 'lenna.jpg'
```

Press ENTER or type command to continue
[1]+ Stopped
```
Who did what

- Albert → Capture Master
- Nektarios → Jpeg Master
- Alex → Ethernet Master
Lessons Learned

- Hardware design is hard
- You cannot debug without simulators
- Jpeg encoding is really tricky
- UDP packets should have checksum and minimum size
- Refreshed our Verilog and gained new VHDL knowledge
- Murphy’s law