Facial Recognition

Alex Newcomb, Tom Stefanyk

Group Members

Alex Newcomb

- In charge of web server, image compression and facial recognition database
- The tall one

Tom Stefanyk

- In charge of input, face detection and face recognition engine
- Not the tall one

Facial Recognition

Unknown Person Detected

O Unlock Door

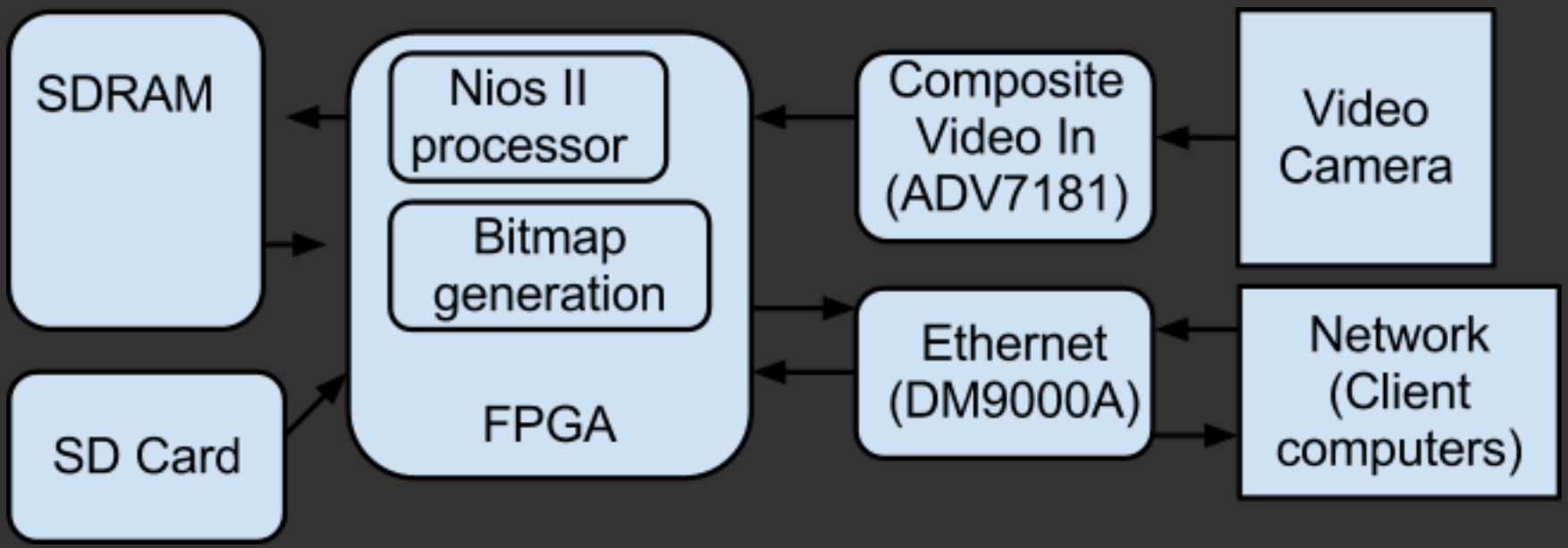
O Send Greeting

O Release the Hounds

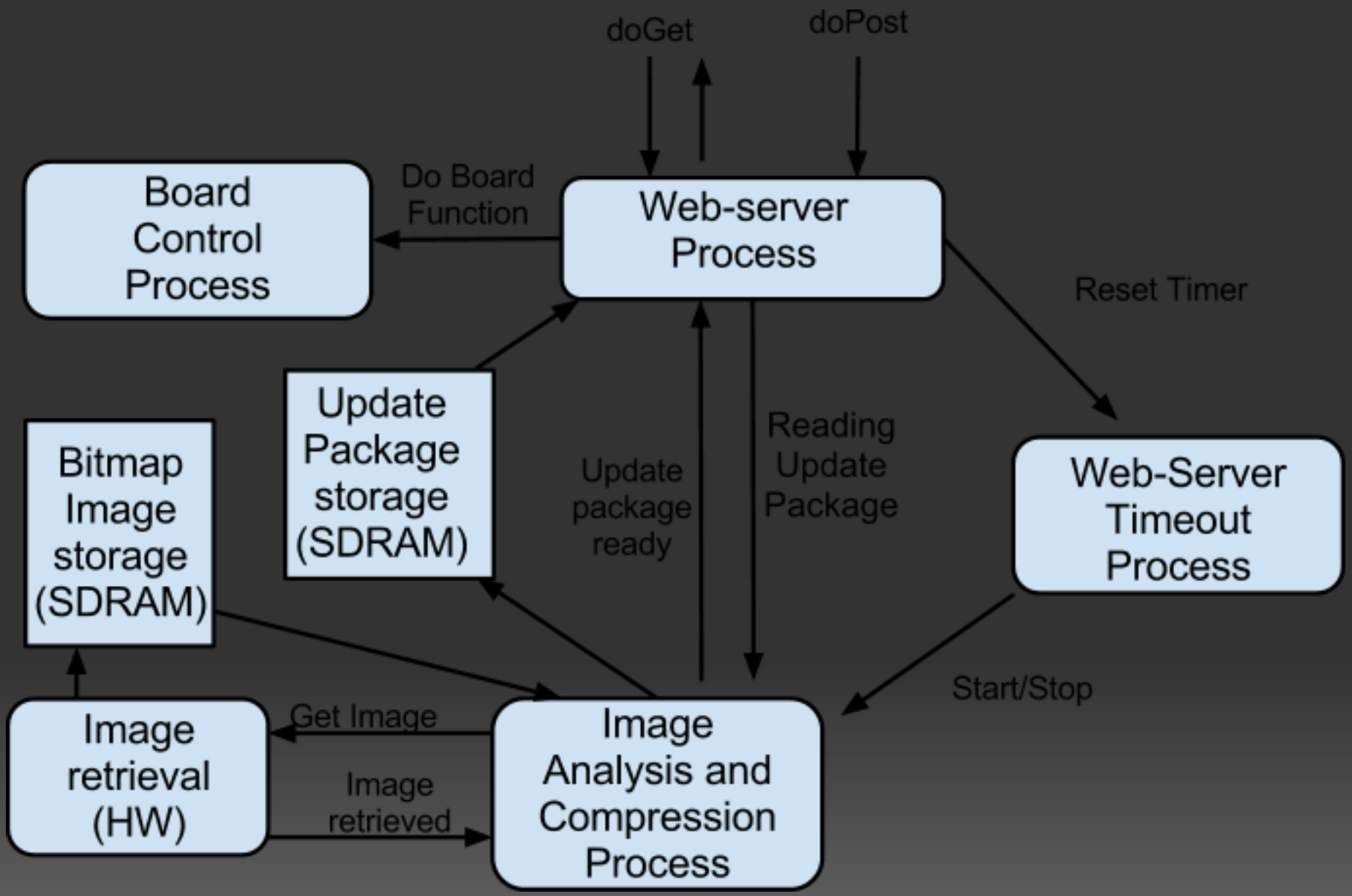
- Analyze image for facial features
- Compare relative position to base
- Identify from database
- Present Image and Actions

image from http://www.engadget.com/2011/08/04/germany-challenges-facebook-on-facial-recognition-citing-eu-pri/

Hardware


SDRAM
SD Card
Nios II processor
Bitmap generation
FPGA
Composite Video In (ADV7181)
Ethernet (DM9000A)
Video Camera
Network (Client computers)

Software


doGet
doPost
Board Control Process
Do Board Function
Web-server Process
Reset Timer
Update Package storage (SDRAM)
Bitmap Image storage (SDRAM)
Update package ready
Reading Update Package
Web-Server Timeout Process
Start/Stop
Image retrieval (HW)
Get Image
Image retrieved
Image Analysis and Compression Process

Challenges

- Updating and modifying given example code
- Determining required memory space to use OpenCV library (either the C or C++ functions)

Components

FPGA-based:
- Video In controller
- DM9000 Ethernet controller interface
- (tentative) On-Chip RAM

On-board:
- 8MB SDRAM
- (tentative) SD Card Reader

Off-board:
- Video camera
- (backup) Wiznet WIZ830MJ Ethernet controller

Code Example

```
int feature_detection(char* name, CvSeq* storage){
//Searches for a given feature within the captured image
//name: Name of an xml file used to search for file
//storage: allocated storage that will hold the locations of
//features, should they be found
//Output: Returns number of features found, or -1 if error

CvHaarClassifierCascade* cascade;                          //Pointer to memory holding patterns to search for in image
const CvArr* image = GREYSCALE_IMAGE_BASE;                  //Pointer to 8-bit greyscale image
const CvArr* orig_img = COLOR_IMAGE_BASE;                   //Pointer to 24-bit color image
CvMemStorage* calculations = cvCreateMemStorage(0);    //Memory space used for calculations
CvPoint pt1, pt2;    //Cartesian point objects

//0. First, make some room for memory
cascade = (CvHaarClassifierCascade*) malloc(SIZE_OF_XML_FILE);
if (cascade==NULL){
printf("Malloc error: unable to allocate space for cascade\n");
return -1;
}

//1. Load cascade into memory
if((CvHaarClassifierCascade*)cvLoad(name, cascade, NULL, NULL)==NULL){
printf("Error loading %s\n", name);
return -1;
}
```

Code Example (cont.)

```
        //2. Equalize image and increase contrast
equalizeHist(image, image);
//3. Get to detecting
cvHaarDetectObjects(image, cascade, storage, SCALE_FACTOR, NEIGHBORS, FLAGS,
                                        cvSize(MIN_X,MIN_Y), cvSize(MAX_X,MAX_Y));
        cvClearMemStorage(storage);


        //4. Release cascade to free up memory
cvReleaseHaarClassifierCascade(&cascade);
//5. Draw the rectangle surrounding an identified feature
for (int i=0;i<storage->total;i++){
CvRect* r = (CvRect*)cvGetSeqElem(faces, i);
pt1.x = r->x;
pt2.x = r->x+r->width;
pt1.y = r->y;
pt2.y = r->y+r->height;
cvRectangle(orig_img, pt1, pt2, CV_RGB(255,0,0), 3, 8, 0);
}

return storage->total;
}
```

Test Plan

Software:
Image analysis:
Test of facial recognition in test pictures uploaded into flash using Nios II utilities.

Compression:
Test of compression on pictures uploaded into flash using Nios II utilities. Results will be saved in flash and downloaded using Nios II utilties for verification.
Save image in update package storage.

Web-Server:
1) Serve sample web-page
2) Serve sample image in doGet response
3) Respond to doPost requests with sample reply
4) Serve image from Update Package Storage

Web-Server Timeout:
Test that the countdown timer generates the correct signal response.
Board Control:
Test that each function results in the desired response from the board.

Hardware:
Ethernet:
Test that connection detection is working (i.e. lights turn on with cable is plugged in)
Test that TCP/IP communication is working (ping responses are correct).

Image Retrieval:
Test that video input signal is being detected (light LED).
Test that image is being saved (push button triggered). Save image into flash and read using Nios II utilities.

App Notes

Instructions for building a web-server using the built-in ethernet card

Includes VHDL interface, C drivers and example top-level

Works with sample Web-Server application

Uses Nios II Software Build Tools for Eclipse

First version is up at http://www.ece.ualberta.ca/~elliott/cmpe490/appnotes/2012w/Webserver/

Other Features

Features to add, given time:

- Eigenfaces method of facial recognition (more advanced and more reliable method)
- Image modification centered on identified facial landmarks

Features to remove, given lack of time

- Implementation of JPEG compression
- Separate storage of grayscale and color images