.

Notes for the System Level Integration Practical

D. K. Arvind[1]
Institute for Computing Systems Architecture
Division of Informatics
The University of Edinburgh
Mayfield Road, Edinburgh EH9 3JZ
Scotland

This document is intended as a reference for the SLI Practical. It describes the specification of a case-study and a description of the ARM-based TIGER Integration Platform

[1] dka@dcs.ed.ac.uk

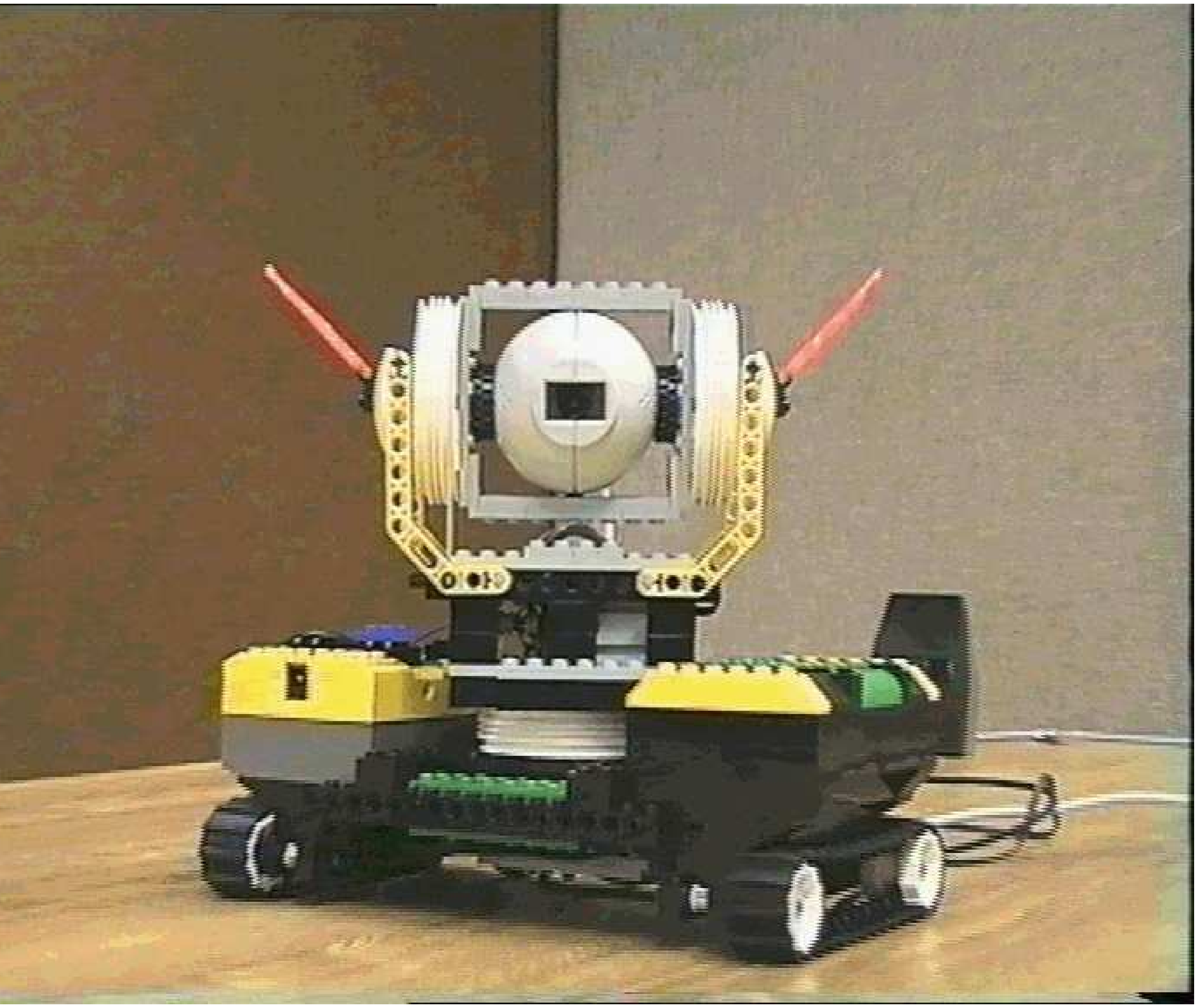

Figure 1: The DRAGON

1 Specification of the Reference Design

1.1 Selection and Analysis of an Application Domain

We analysed a number of applications (see Table ??) which could be classified broadly under the *multimedia* category. The aim of the exercise was to identify a set of applications which could be implemented on integration platforms. In particular, we wished to identify a core set of hardware and software functions which would constitute the foundation block of such a platform - the idea being that the foundation block will be invariant part and reused in different platform designs.

Application	Multimedia	Mobile
Non-satellite GPS	X	X
Satellite movie distribution	X	
Guardian Angel		X
Real-time animatronics	X	
Electronic Pet	X	X
Tapeless video camera	X	X
Electronic touch judging	X	
Pulse Radio	X	X
Tracking camera for videoconferencing	X	
Digital still camera	X	X
Video camera with live uplink	X	X
- Medical applications, TV		
- Probes for remote survey		

Table 1: Multimedia applications

The rest of this section describes the behaviour of the reference and derivative designs and their *Requirements and Use Case Analysis*.

1.2 Requirements for Tracking Camera for Video-Conferencing

The intended use of the tracking camera is in video conferencing. The system has two purposes: to recognise the faces of people participating in the video conference; to track the face of one of the participants, perhaps zooming and swivelling the camera to follow this face. The components and their descriptions are described next.

Component - The camera grabs frames at a rate of 25 frames per second. Each frame is sent in digital form to the controller over a parallel connection.

Controller - The controller then does the following:

- Identify regions of the frame containing faces.
- Track the position of a face. This may involve zooming and swivelling the camera.

Camera control - This controls the zoom factor and orientation of the camera.

PC - The PC receives images and face identification information from the controller.

1.2.1 Functional Requirements

Face Detection - High Priority, Medium Risk
The controller identifies rectangular regions of the frame that potentially contain faces. The output sent to the PC includes the location of the faces within the frame, and the corresponding cropped image. It should complete this function within four seconds. This feature has a high priority because the other features of the controller depend on it.

Face Recognition - High Priority, Medium Risk
The controller maintains a database of faces that it can recognise. Records may be added, modified and deleted. The user will perform these operations using a GUI. The controller identifies recognised faces among those found during Face Detection. It should inform the PC of all recognised faces within five seconds. Much of the software for face recognition is already developed and has been found to meet the time constraints given above, so this feature has medium risk.

Face Tracking - High Priority, High Risk
Once a face has been detected, the controller can track it in real time. The controller outputs the rectangular region of the frame containing the face and the corresponding cropped image. It updates at least 15 times per second. It may also issue commands to the zoom control and swivel stand to follow the face.
If the controller loses the face it is tracking, then it holds the region at its current position for two seconds. If it does not find the face in this region within this time, then it resets the camera to its zoomed out state and attempts to redetect the face.

GUI - High Priority, Low Risk
The PC displays a rectangular region of the frame and identification information. It also allows the recognition database to be maintained. No other requirements are identified at this time.

1.3 Constraints

The camera grabs 25 frames per second. Each frame is 160 by 120 pixels. The pixels are 8-bit greyscale. The frames are transferred over a parallel or USB connection. The camera my be zoomed or swivelled. The controller will be implemented as an embedded system, using 0.35μm CMOS technology. An ARM7TDMI but without floating point operations is available. The pSOS real-time operating system and the PRISM+ developmental environment is available. Other components such as DSPs, and JPEG compressors may be used. The components will communicate over an Amba bus.

1.4 Use Case Analysis

1.4.1 Actors

Observer - The participant in the session that is viewing subjects

Maintainer - A user who maintains the recognition database.

Camera - The frame grabbing device

Camera control - The device that controls camera zoom and swivel

Display - The device that displays information for the observer

1.4.2 Use Cases

Reset

Actors involved - Observer, camera, camera control, display

Precondition - The system is in any mode.

Basic case -

1. The observer instructs the system to reset.
2. The camera, display and camera control are reset.
3. The camera, display and camera control are tested.

Alternative cases - If any device is not working, an error is reported and the system is put in the error mode.

Postcondition - On normal completion, the system is in the idle mode. Otherwise it is in the error mode.

Begin Detection

Actors involved - Observer

Precondition - The system is in the idle mode.

Basic case -

1. The observer instructs the system to begin detecting or recognising.
2. Put the system in the detecting or recognising mode

Postcondition - The system is in the detecting or recognising mode.

Set tracked subject

Actors involved - Observer

Precondition - The system is in the detecting or recognising mode.

Basic case -

1. The observer instructs the system to track a particular subject.
2. Put the system in the tracking mode.

Postcondition - The system is in the tracking mode.

Detect subjects

Actors involved - Camera, Display

Precondition - The system is in the detecting mode.

Basic case -

1. The camera provides the next frame.
2. Report the location of these subjects.

Postcondition - The system is in the detecting mode.

Constraints - Must complete within 4 seconds.

Recognise subjects

Actors involved - Camera, Display

Precondition - The system is in the recognising mode.

Basic case -

1. The camera provides the next frame.
2. Recognise subjects.
3. Report the location of these subjects.

Postcondition - The system is in the recognising mode.

Constraints - Must complete within 5 seconds.

Track subject

Actors involved - Camera, Display, Camera control

Precondition - The system is in the tracking mode.

Basic case -

1. The camera provides the next frame.
2. Determine current location of tracked subject.
3. Report the current location of the subject.
4. Optionally, instruct the camera to follow the subject.

Alternative cases - If the subject cannot be located then put the system in the lost mode.

Postcondition - On normal completion, the system is in the tracking mode. If the subject is lost, then the system is in the lost mode.

Constraints - Must complete within 1/15 second.

Lost tracking

Actors involved - Camera, Display

Precondition - The system is in the lost mode.

Basic case -

1. The camera provides the next frame.
2. Attempt to determine current location of tracked subject.
3. If successful, then return to tracking mode.

Alternative cases -

1. If unsuccessful after two seconds, then go to idle mode.
2. Otherwise, remain in lost mode.

Postcondition - The system is either in the lost, idle or tracking mode.

Constraints - Must complete within 1/15 second.

Update recognition database

Actors involved - Maintainer

Precondition - The system is not in the error mode.

Basic case - The maintainer adds, modifies or removes a record from the recognition database.

Postcondition - The system is in the same mode as on entry.

1.5 Requirements Analysis of the Derivative Design

1.5.1 Requirements for an Electronic Gate Keeper for the visually challenged

A visitor ringing the bell triggers the face detection and recognition modules. If the face is matched against a dictionary of known faces, then the name is output (through a speech synthesizer, for instance), and the person can take appropriate action. Faces which are not recognised are compressed and stored for future inspection and a designated person is alerted. The components and their descriptions are described next.

Bell - A visitor ringing the bell triggers the contoller

Camera - The camera grabs frames at rate of 25 frames per second. The digitised frame is sent to the controller over a parallel connection.

Controller - The controller performs the following tasks:

- Identify regions of the frame containing faces and take snapshots.
- Zoom and swivel the camera to track the faces (optional)

Camera control - This controls the zoom factor and orientation of the camera. (optional)

A Persistent memory - A persistent memory device for the images

1.5.2 Functional Requirements

Face Recognition - High Priority, Medium Risk

A database of faces is stored in the persistent memory. Records may be added, modified and deleted. The user will perform these operations using a GUI. The bell triggers the face detection and informs the user should a face be recognised. Otherwise, it records a snapshot of the face in the persistent memory for future inspection. The user will view these snapshots using a GUI. Face recognition must take no longer than 5 seconds.

Much of the software for face recognition is already developed and has been found to meet the time constraints given above, so this feature has medium risk.

GUI - High Priority, Low Risk

The GUI provides two features. Firstly, it allows the faces database to be maintained. Secondly, it allows snapshots of those faces not recognised to be viewed. No other requirements are identified at this time.

Constraints - The camera grabs 25 frames per second. Each frame is 160 by 120 pixels. The pixels are 8-bit grey scale. The frames are transferred over a parallel or USB connection. The camera my be zoomed or swivelled. The controller will be implemented as an embedded system, using 0.35μm CMOS technology. We intend using an ARM7TDMI processor, but without floating-point operations. The pSOS real-time operating system will be used. Other components such as DSPs and JPEG compressors may be used. The components will communicate over an Amba bus.

1.6 Use Case Analysis

1.6.1 Actors

Resident - a visually impaired person staying in the house

Camera - the frame grabbing device

Camera control - the device that controls camera zoom and swivel

Persistent Memory - The persistent memory device that stores compressed images of unrecognised faces

Visitor - the person who rings the bell at the door

Maintainer - the person who maintains the face database

1.7 Use Cases

1.7.1 Recognise visitors

Actors involved - **Visitor**, Camera,

Precondition - Visitor rings the bell

Basic case -

1. The camera provides the next frame
2. If one or more faces are detected and recognised, then signal the resident with the name(s)

Alternative case - If the face is not recognised then signal “unrecognised” and store a shapshot of the face

Constraints - must complete within 5 seconds

2 Implementation of the PC Prototype

This section describes the implementation of the Reference Design on a PC running Windows NT 4.0 (Servicepack 3). The application was implemented in C++. Much of the software was developed initially on a SUN UltraSparc 10 workstation running Solaris. Apart from a few endianess issues the porting of the code did not pose too many problems. The Reference Design can be divided into the following six modules:

Control unit - The control unit module implements the following behaviour: the *face detection* module checks for the presence of faces in the video frames from the camera. On finding a face, the *face recognition* module compares it to a reference face; if positive, the *tracking module* tracks this face across subsequent frames. The camera stand interface turns the camera to ensure that the face remains in the central region of the frame.

Face Detector - This module scans the image by feeding 20 X 20 pixel subimages into a pretrained artificial neural network (ANN). The network consists of approximately 500 nodes and 5000 weighted connections. The network is evaluated for each part of the image which involves the multiplication of the nodal value with the weights and the accumulation of the results. One evaluation of the network involves 5000 multiply-accumulate operations.

Face Recogniser - This module is implemented as a fully-connected, 3-layered back propagation network. The input layer consists of 1024 nodes representing a 32 X 32 pixel input image. The middle layer has 32 nodes and the output layer has 1 node. The input to the network is the difference between the input and reference images. If the value of the output node is greater than 0.5, then the face is deemed to have been recognised. Given that the network is fully connected, there are 32 X 1024, i.e. 32,768 weights in the first layer, and 32 in the second. One evaluation of the network involves 32,800 multiply-accumulate operations.

Face Tracker - This was implemented using an iterated multidirectional edge detection algorithm.

Frame Grabber - The 160 X 120 pixel frames in 8-bit gray scale are grabbed from a Connectix QuickCam.

Camera control - The camera stand and interface library have been developed using the LEGO MindStorms kit which includes a programmable robot control system and software to communicate with the Control unit.

3 System Design in VCC

3.1 Platform Design

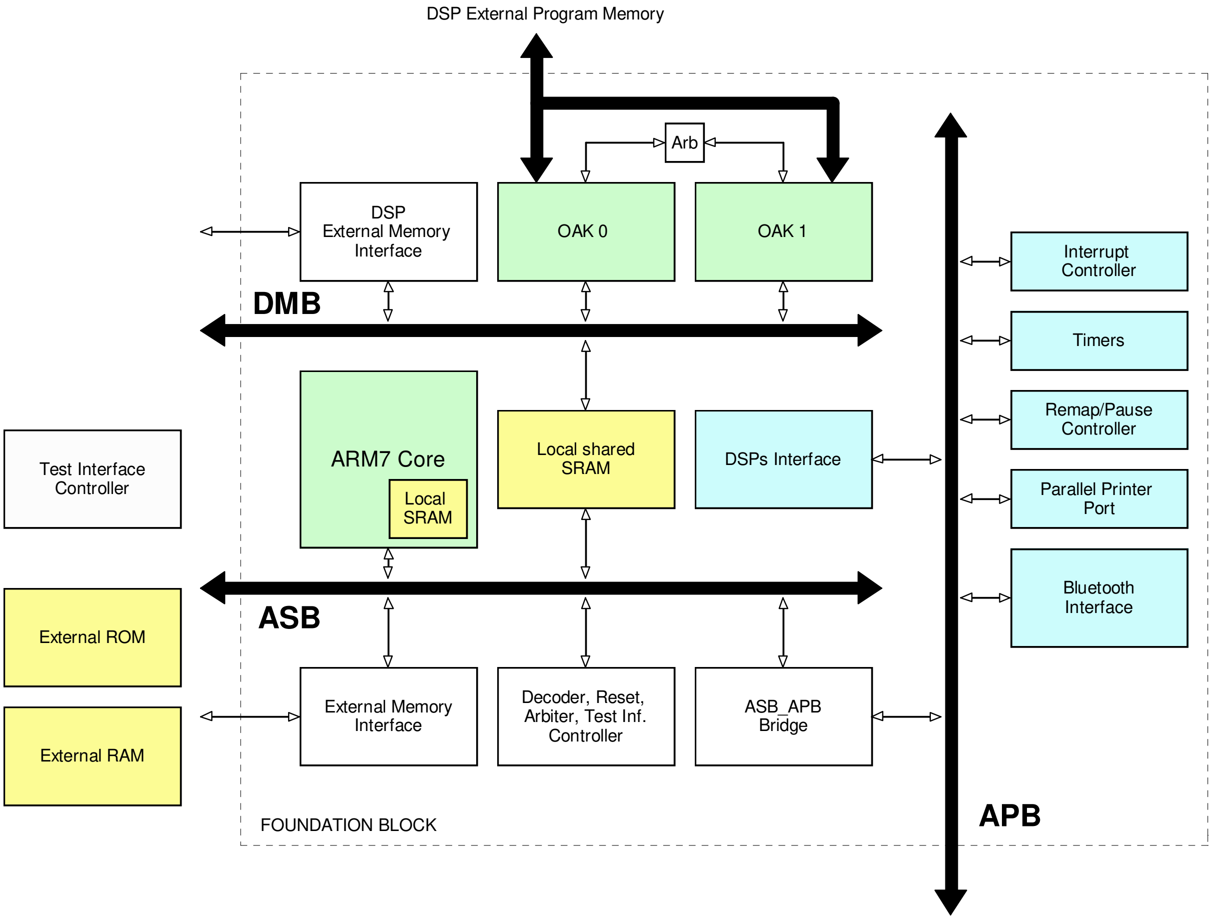

Figure 2: TIGER Integration Platform

This section describes the foundation block architecture as illustrated in Figure ??. The design incorporates an ARM processor and multiple OAK DSP cores. The two subsystems communicate via a region of shared (dual-ported) memory.

The TIGER platform has a SIMD array of DSPs. These DSPs access a stream of neural network data through a register in the DSP memory controller. The DSPs simultaneously access the register and receive the next word in the stream.

The DSPs are supported by a broadcast bus mode and memory controller which both extend their addressing range and offers autonomous reading and buffering of external memory (stream buffer).

The ARM system contains an ARM7TDMI core supported by AMBA system and peripheral buses.

3.2 Foundation Block (FB) components

As mentioned above the AMBA blocks from the Micropack are implemented in VHDL and have been modified for our platform design. These are described next.

- ARM7TDMI.
  The ARM processor is the main master in the architecture and is responsible for the overall control of the system. It carries out different routines, from the top-level applications such as face traction to the distribution of tasks, such as face detection and recognition, to the DSP subsystem.

- Arbiter.
  It provides arbitration between the master components of the FB for access to slaves within the ASB bus. The arbiter can handle up to four masters in the current implementation.

- Decoder.
  The decoder selects a slave from among the different targets based on the address that the active master provides once it has started a transaction. The decoder can be easily modified to add more targets by increasing the signal `DSELx` and decoding their addresses accordingly. At present the decoder selects between internal and external memory, and between the ASM and APB buses.

- ASB/APB Bridge.
  If the peripheral bus is selected by the decoder, then the bridge coordinates the transfers between the ASB and APB buses. As the components of the peripheral bus may be slow, they should not have to use the global clock, to save power, given that they may not be operating at all times.

- Reset Controller.
  The purpose of the reset controller is to provide a “cold” reset to the system. An external power-on reset triggers what it is the main reset (`BnRES`) of the chip. The ARM, for example, needs a few cycles with the reset being *low*, in order to exercise dummy instruction fetches.

- Static Memory Interface
  The static memory interface (SMI) serves as a synthesisable behavioural memory block that can be configured for different memory wait states. This block should only be instantiated in order to test the system. It would eventually be replaced by an *External Bus Interface* which would take into account process and package dependent delays.

- Memory.
  The external ROM consists of 4 banks of 32K x 8 bits and is where `Virtual ICE` maps the application code starting at address 0x00000000[2]. The internal RAM is a 32 bit little-endian, 32 K byte RAM with single wait state. However, it does not represent a truly on-chip SRAM memory. The external memory is a writable copy of the ROM and the size is also 32 Kbytes.

[2] This is explained in more detail in Section **??**.

- Interrupt Controller.
  The interrupt controller, located in the APB bus, gathers all the interrupt requests from the different devices and grants the one with the highest priority. It raises one of two global interrupt signals that are monitored by the ARM. One is called *Fast Interrupt Request* `FIQ` (that is a high priority), low latency handling interrupt and a slower Interrupt Request (`IRQ`).

  The components capable of requesting an interrupt are the Timers, the Remap/Pause Controller, the JPEG and the DSP controller. These are explained later.

- Timers
  They consist of two counters that can be used in a free-running or periodic timer modes. In the free-running mode the timer wraps after reaching its zero value and continues to count down from its maximum value. In the periodic mode, the counter generates an interrupt at a previously-set constant interval.

- Remap/Pause Controller
  The remap and pause controller represents the combination of four functions: the *pause* function covers the mode in which the system can be in a low power function; *identification* gives an indication of the system configuration; *reset memory map* can be used to change the low-memory mapping under reset conditions[3], and *reset status* provides an indication of the cause of the last reset condition.

- OAK-based DSP Bank
  The bank of OAK DSPs resides in the DMB bus and is designed to expand up to eight OAKs, of which only two are situated in the Foundation Block. They are targeted for face detection and recognition, respectively. The additional OAK in the derivative is designed to run the face detection in parallel with the one in the FB.

- OAK Arbiter
  The OAK arbiter is responsible for providing fair arbitration to the OAKs when accessing the DMB bus.

- DSP Memory Controller
  The DSP memory controller processes the different transactions from the OAKs once they have been granted the bus. It handles read and write signals for both the external and shared SRAM.

- Dual Port Shared SRAM
  The shared SRAM is a dual port 16 Kbytes RAM where reads and writes from the ASB and DMB buses can occur concurrently. The case of both ends writing to the same location is not considered because of the nature of the architecture; the synchronisation between the ARM and the OAKs would rule out such cases.

- SRAM Decoder
  It detects when the SRAM is being accessed from the ASB bus and enables the SRAM. It also converts the control signals from an AMBA specification to a low-active RAM type.

[3] This function has been disabled since `Virtual ICE` has to map the application code (linked from C and assembler code) to the ROM at startup. This function sets in conflict that operation, i.e. the ARM would not find the start-up code.

- DSP Interface
  This is responsible for the synchronisation between the ARM and the bank of OAKs. The ARM always starts the communication by writing to a register in the DSP interface. Once this is done, the interface interrupts the OAKs so that they can start their respective processes. As soon as they finish, each of them sends a completion signal and will wait until all of them have finished. The DSP interface then interrupts the ARM in order to gather results or to provide more data to the OAKs.

- JPEG Encoder
  This is an in-house design from the previous evaluation. Its input and output buffers have been customised in order to accept 20 blocks of data (20 x 64 bytes of grayscale pictures). The output buffer has also been modified to generate an interrupt as soon as it covers all the data.

3.3 Memory Maps and I/O

3.3.1 ARM System

Address Range	**Use**
0x0A000000-0x0BFFFFFF	APB Slaves
0x08000000-0x09FFFFFF	ASB Expansion Slots
0x04000000-0x07FFFFFF	ROM
0x03000000-0x03007FFF	Shared RAM
0x00000000-0x03FFFFFF	RAM

Table 2: Main Memory MAP

Address	**Use**
0x0A000000	Interrupt Controller
0x0A100000	Timer
0x0A200000	Remap and Pause Controller
0x0A300000	Camera Interface (Parallel Port)
0x0A400000	DSP Interface

Table 3: APB Memory MAP

A standard programmer's model of ARM based systems is given in ARM's Reference Peripheral Specification. This defines standard offsets for registers in a number of peripheral memory maps. Additional information specific to the foundation block is given below.

3.3.2 Interrupt Controller

Interrupt controller registers are as defined in [?]. Interrupt controller defined bits 7 and 8 are reserved for the DSP and JPEG, respectively.

3.3.3 Camera Interface

Address	**Read Location**	**Write Location**
CameraBase + 0x00	Frame Time (ms)	Grab Frame
CameraBase + 0x04	Frame data (8-bit)	Clear IRQ

Table 4: Camera Interface Registers

3.3.4 DSP Interface

Address	**Read Location**	**Write Location**
DSPBase + 0x00	Interrupt Status	Interrupt Control
DSPBase + 0x04	DSP Clock Control	DSP Status

Table 5: DSP Interface Registers

3.4 OAK System

The OAK DSP's have 2K of both X and Y on-core RAM, indicated by holding the pin `OMEMSZ4P` high.

Address Range	**Size**	**Use**
0xF800-0xFFFF	2k	YRAM
0xC000-0xF800	14k	UNUSED
0x8000-0xBFFF	16k	External Memory
0x4000-0x7FFF	16k	Shared Memory
0x07FF-0x3FFF	14k	Reserved (for local XRAM)
0x0000-0x07FF	2k	XRAM

Table 6: OAK Main Memory Map

The two MSB's of the memory address control which memory is accessed (local, shared or external).

3.4.1 User Defined I/O

The OAK's user controllable I/O pins are used to implement the arbitrated bus scheme and to allow a switch between arbitrated and broadcast bus modes. The pins are described in Table **??**.

Input pins can be tested under software control (bits 11,10 of $ST2$ register - $IU1$, $IU0$). Output pins can be set under software control (bits 9,8 of $ST2$ register - $OU1$, $OU0$).

Address	Interrupt
0x0016	INTerrupt2
0x000E	INTerrupt1
0x0006	INTerrupt0
0x0004	NMI
0x0002	TRAP/BI
0x0000	Reset

Table 7: OAK Program Memory - Interrupt Vectors

Pin	Description
BIUSER0P	Bus Grant
BIUSER1P	UNDEFINED
CUSER0P	Bus Request
CUSERO1P	Set Completion Bit

Table 8: OAK User I/O Pins

3.4.2 OAK External Register Configuration

The buses $PEDSTN$ (0-5) and $PESRCN$ (0-5) indicate the destination and source operand in the current instruction. The $PRWEXTP$ pin indicates a read/write from/to external user defined registers.

All DSP external registers are mapped to the memory controller as shown in Table **??**.

Code	Register	Read Location	Write Location
010100	ext0	Undefined	Upper memory address word
010101	ext1	Undefined	Lower memory address word (for stream buffer)
010110	ext2	Control/status register	Control/status register
010111	ext3	Head of stream buffer	Undefined

Table 9: OAK External Register Configuration

The external memory address is generated by concatenating the lower 14-bits of the DSP address bus with the lower 8-bits of the upper memory address word (memory controller). This creates a 22-bit address allowing up to 4Mb of external memory to be addressed.

External memory wait states are programmable within the range 0 to 7. The value is set by writing to the memory controller's status/control register (bits 12-10 as defined in Table **??**.

The memory controller contains address generation logic and a buffer for reading sequential external memory locations. The head of the buffer may be read by the OAK in a single cycle through external register $ext3$.

The memory controller will begin to read from external memory into the stream buffer

Bits	Use
15-13	Unique identifier for each DSP
12-10	External Memory Wait States
9	Reset Stream Buffer
8	UNDEFINED
7-0	DSP Bus Request Signals

Table 10: Memory Controller Control/Status Register

as soon as a value is written into the OAK's external register $ext2$. The read process may be halted by setting bit 9 of the memory controller's status/control register. This bit must be cleared and a new address written to $ext2$ before reading will resume.

The memory controller will hold *bwait* high while reading new data if the stream buffer is empty when a read request is received.

3.5 OAK DSP Bus Access Modes

Here are examples of the DSPs operating in both arbitrated and broadcast bus modes under the control of the ARM processor.

3.5.1 Arbitrated Bus Mode

1. ARM sets up shared memory area
2. ARM writes to control register in DSP interface (Interrupt Control)
3. DSP Interface interrupts OAK DSPs
4. OAK DSPs create bus request signals (See Section **??**)
5. Arbitrated Bus mode - Each DSP has access to bus in turn.
6. Each DSP sets completion bit (in DSP controller control register)
7. When all bits are set ARM receives interrupt from DSP controller

3.5.2 Broadcast Bus Mode

If no bus request is made and the DSP memory bus is accessed then all the DSPs will receive the same data across the bus. Appropriate address bus tristating is implemented for all DSPs apart from the designated address master. ARM/DSP interrupts may occur as described above.