TAYRA—A 3D Graphics Raster Processor

Marcus Waller, Graham Dunnett, Mike Bassett, Shaun M^{c}Cann,
Alex Makris, Martin White, Paul Lister
Centre for VLSI and Computer Graphics

Abstract

This paper describes the functionality of a 3D Graphics Raster Processor called TAYRA. TAYRA consists in the most part of Graphics Raster Pipeline with five major external interfaces: PCI Master/Target, Depth, Texture, Colour and Video Interfaces.

The Graphics Raster Pipeline performs all the major OpenGL style (not necessarily compliant) raster functions: scan conversion; lines, spans, triangles, rectangles, perspective correction of texture coordinates, mip map level selection, and many other texture modes, alpha blending, and other functionalities. Further, through TAYRA's fast host to buffer access mechanisms it can do advanced stencilling, multi-pass antialiasing, and other algorithms; all accelerated in hardware with a sustained pixel write speed of 29 MPixels/sec (peak of 33 MPixels/sec). This translates to a peak 25 pixel triangle drawing speed of 890K Triangles/sec, limited by PCI bus bandwidth.

1. Introduction

TAYRA is a 3D Graphics Rasteriser Processor that is specified to compete in the high end 3D graphics application markets on PCs, and the arcade video games market. To penetrate this market effectively it has been determined that TAYRA should have a large range of 3D functionalities that exceeds the perceived competition, but at a lower price versus performance ratio. Accordingly, TAYRA is specified with a fully compliant PCI Local Bus Specification Revision 2.1 Interface, and other interfaces to 3D graphics depth, texture and colour buffers.

In the following sections we describe TAYRA's functional requirements or features. That is, what TAYRA will do, not how it does it. More than a brief overview of TAYRA's architecture is beyond the scope of this paper. The reader is invited to access TAYRA's web page via: *http://www.sussex.ac.uk/engg/research/vlsi/*, over the coming months to see more detailed descriptions of TAYRA's architecture components, external interface definition and programming specification.

2. Feature Summary

TAYRA is based fundamentally on algorithms found in standard texts and research papers on computer graphics with novel features and implementation techniques. TAYRA's scan conversion algorithm uses linear edge functions similar to Pineda [1], but with special look ahead logic [2] used to determine in advance the next pixel during scan conversion. Also pixel clipping is incorporated as part of the scan conversion. Floating point number representations are used where the dynamic range of numerical data can vary greatly, especially in the rational linear interpolation of texture coordinates.

Table 2-1 compares TAYRA's features with that of the Lockheed Martin R3D/100 chipset. This serves to illustrate not only TAYRA's impressive features but also its high level of integration compared with the R3D/100.

Features	TAYRA	R3D/100
32 bit PCI Local Bus Interface	R2.1	R2.0
High Performance Communication FIFOs	✓	✓
Scan Conversion of Lines, Spans, Triangles and Rectangles	✓	✓
Integrated Depth Test and Buffer Controller	✓	✗
OpenGL Fogging	✓	✗
Fast Context Switching Program Register Set	✓	✗
Colour Buffer Controller	RGBA	RGB
True Colour Double Buffer (Up to 16 bit, 1600x1200)	✓	24 Bit
Display Resolution (320x240 to 1600x1200)	✓	✗
Depth Buffer Controller	32	24 bit
True Floating Point Perspective Correction	FLP	FXP
Hardware Mip Mapped Level of Detail	✓	✗
32 Bit RGBA Texture Mip Mapping	✓	✗
Square, Linear, Point2D Mip Map Support	✓	Square
Wrap (W), Invert (In), Ignore (Ig), Clamp (C) Texture Tiling	W, In, Ig, C	W, C
Texture Colour Map Format Modes	✓	✗
OpenGL Texture Decal and Modulate	✓	?
32 Bit RGBA Texture Point, Linear, Bilinear, Trilinear Filtering	✓	✗
OpenGL Blending (Transparency, Antialiasing)	✓	✓
Multi-pass Antialiasing	✓	✓
OpenGL Stencilling	✓	✓
Clipping to regions	✓	✓
Host to Depth, Texture, Colour Buffer Access	✓	✗
Fast 8.5 GByte/sec Depth Buffer Clear Operation	✓	?
Fast 8.5 GByte/sec Colour Buffer Clear Operation	✓	?
Depth Buffer support (TAYRA— SGRAM or SDRAM)	8 MBytes	5 MBytes
Texture Buffer support (TAYRA—SGRAM or SDRAM)	16 MBytes	8 MBytes
Colour Buffer support (TAYRA—VRAM or WRAM)	8 MBytes	10 MBytes
Global Depth, Texture, Colour Refresh Controller	✓	✗

Table 2-1 Comparison of TAYRA and R3D/100

3. Performance

Table 3-1 compares the performance figures of TAYRA versus the Lockheed Martin R3D/100 chip set [3]. The Lockheed Martin R3D/100 chip set consists of about six chips including a 32 bit floating point processor. TAYRA is a fully integrated single chip solution, needing only the host processor for Geometry and Setup computations.

TAYRA can achieve the performance figures given in Table 3-1 because of the highly efficient memory controller designs for the depth, texture and colour buffers. For the depth and texture buffers we use the latest SGRAM and SDRAM technologies, for which we have designed memory controllers that make optimal use of page mode cycles. Although, in general for texture we can not exploit the coherency that depth exhibits, we can arrange texture maps in memory to make the best use of page mode cycles. The colour memory controller is the worst case bottleneck. This is because it is very difficult (within reasonable design constraints) to perform a *read, modify, write* in a single graphics pipeline clock cycle, we can of course still do *read, modify, write* but at a significant performance decrease. However, we can achieve the above performance figures for write only operations. These operations would include transparency where the current pixel is blended with an on-chip background colour, antialiasing with background colour, transparent textures, etc. Also, we should consider that not all objects are alpha blended in a typical scene. For example, an illuminated, shaded and textured scene represents an high quality image. This can be rendered at the above rates requiring only colour writes.

The performance figures above are based on a 33 MHz graphics pipeline clock and 66 MHz memory controller clocks. However, we are still PCI input bandwidth limited to a peak 890 KTriangles/sec. A future solution to this bottleneck could be the Accelerated Graphics Port (AGP) architecture. AGP is an extension to the PCI architecture which adds a demultiplexed address bus, pipelined transfers and 133 MHz transfer rates to boost graphics performance. Another solution is to integrate a setup engine and vertex level interface to allow triangle strips to be processed. This effectively decreases primitive vertex data crossing the PCI from three vertices per triangle to about one.

Performance Equivalents		
Pixel Writes	**25 Pixels/Triangle**	**50 Pixels/Triangle**
TAYRA[1] 3D Triangles *32 bit RGBA Gouraud shaded, 32 bit depth buffered, clipped, stencilled, alpha blended, mip mapped, texture mapped*	890 K/sec	515 K/sec
Lockheed Martin[2] 3D Triangles *32 bit RGBA Gouraud shaded, 24 bit depth buffered, clipped, stencilled, alpha blended, mip mapped, texture mapped*	750 K/sec	375 K/sec
Pixel Writes	**10 Pixels/Line**	
TAYRA[3] Lines *32 bit RGBA Gouraud shaded, 24 bit depth buffered, antialiased*	1.6 M/sec	
Lockheed Martin[2] Lines *32 bit RGBA Gouraud shaded, 24 bit depth buffered, antialiased*	1.5 M/sec	
TAYRA[4] Block Moves *32 bit RGBA and 32 bit Depth*	132 MBytes/sec	
Lockheed Martin[2] Block Moves *32 bit pixels*	132 MBytes/sec	

Table 3-1 TAYRA Peak Performance Comparison

[1] *These performance figures are based on TAYRA's PCI bandwidth and Z-Buffer memory controller. For example, if TAYRA's pipeline is clocked at 33 MHz (66 MHz memory controller) then for a triangle whose pixels exhibit no page faults TAYRA can output 33 MPixels/sec. However, TAYRA has to buffer 12% of a triangles pixels due to Z-Buffer page faults TAYRA, therefore outputs 29 MPixels/sec. Thus, TAYRA can stream 29 MPixels/sec ÷ 25 Pixels/Triangle = 1.16 MTtriangles/sec past the Z-Buffer controller. But an average textured and illuminated triangle at the PCI input consists of 148 Bytes of data loaded by the host. Thus, with a peak PCI bandwidth of 132 MBytes/sec we can see that for 25 Pixel triangles TAYRA is limited by the PCI bandwidth to 132 MBytes/sec ÷ 148 Bytes/Triangle = 890 KTriangles/sec.. Similarly, 50 Pixel triangles are limited by the pipeline bandwidth.*

[2] *Lockheed Martins R3D/100 architecture peak performance figures are quoted from their brochure.*

[3] *Limited by the peak PCI bandwidth of 132 MBytes/sec. A line on average is composed of about 84 bytes of data. Therefore, 132 MBytes/sec ÷ 84 Bytes = 1.6 MLines/sec.*

[4] *TAYRA's block move drawing performance is based on the PCI input bandwidth of 132 MBytes/sec and TAYRA's WRAM colour memory controller output bandwidth of 132 MBytes/sec. The datapath through TAYRA can also sustain 132 MBytes/sec.*

4. TAYRA Architecture

The objectives for TAYRA were to design a high performance 3D Graphics Raster Processor which implements all the common 3D graphics raster functionalities. Further, TAYRA should interface to the latest memory technologies in order to achieve unrivalled performance. All this is to be integrated in a single VLSI package interfaced to a PCI based host. An abstraction of the TAYRA architecture which fulfils these objectives is illustrated in Figure 1.

5. Interface Specification

TAYRA employs five major interfaces to the system environment:

1. PCI Master/Target
2. Depth Buffer
3. Texture Buffer
4. Colour Buffer
5. Video Interface

5.1 PCI Master/Target Interface

TAYRA implements a 32 bit Master/Target PCI Local Bus Revision 2.1 Interface. More detailed descriptions can be found in our PCI data sheet [4]. TAYRA's PCI Master/Target Interface has the following functions:

- Full PCI Revision 2.1 compliant Master/Target Interface
- Full 32 bit data path for internal read and write data
- Four base address registers of 3x256 MByte and 2x4 KByte memory space
- Full optional parity error detection and creation
- High performance FIFO write operation to

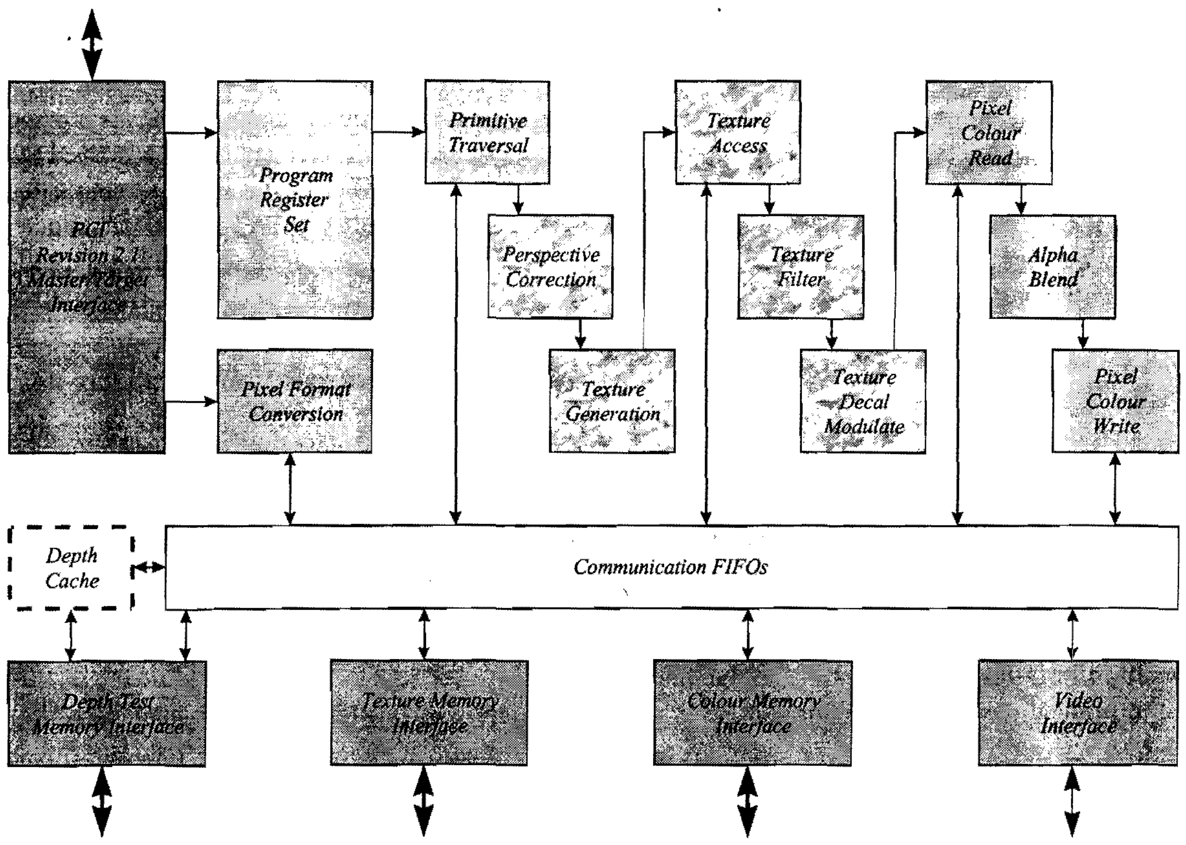

Figure 1 TAYRA Architecture

texture, colour and depth memory interfaces, operating in excess of 120 MBytes/sec

- A further FIFO provides write access to the 3D graphics pipeline at over 120 MBytes/sec
- A low cost read interface provides single word read access to all TAYRA internal registers and TAYRA memory buffer domains
- Special configuration control registers that improve performance for different applications
- Transparent Big and Little Endian access to TAYRA memory domains
- Sequential and context register set primitive ports
- Full function PCI bus mastering primitive list retrieval engine
- Full function PCI bus mastering texture and colour buffer retrieval engine

5.2 Memory Interfaces

TAYRA memory clock frequency is always twice that of the graphics pipeline, which helps to remove the traditional memory access bottleneck. Both the graphics pipeline and the memory systems are clocked independently from the PCI bus which goes towards making the chips performance more independent of the system bus.

TAYRA has three memory interfaces: texture, depth and colour, giving complete independence between the access protocol used by each and enabling a dedicated and optimised memory system to be configured for each system's particular requirements. It also means that the latest memory technology can be employed. For example, the advanced block write feature of SGRAM is useful for buffer clear operations in the depth memory and the advanced BitBLT capabilities and block write functions of the new Window RAM (WRAM) can be employed to best advantage in the colour buffer. The dual port capability of the WRAM can also be used to optimise the display of the colour data.

5.2.1 Depth Buffer Interface

TAYRA implements a programmable generic depth buffer memory controller, which will support both SGRAM and SDRAM, for accessing the local depth buffer. This memory controller will arbitrate between the host (via TAYRA's PCI interface through the on-chip communication FIFOs), and the graphics pipeline.

The depth buffer Interface has the following features:

- Intelligent arbitration scheme designed to support the advantages of PCI and memory burst mode operation
- Built-in depth test for fast read, compare, write cycles at graphics pipeline speed
- 264 MByte/sec read/write bandwidth performance at 66 MHz
- 66 MHz memory clock frequency nominal
- Memory operations supported include read, write, super page mode buffer clear at 8.5 GBytes/sec and 2.125 GBytes/sec for SGRAM and SDRAM respectively (8 memory chips), masked write, and proprietary block write functions
- Supports from 1 to 8 MBytes of SGRAM or SDRAM
- Global refresh controller.
- Memory access latency is absorbed by FIFOs
- Small FIFO in the memory controller absorbs read-modify-write access dependencies

Some memory planes of the depth buffer can be configured for use as stencil planes.

5.2.2 Texture Buffer Interface

TAYRA implements a programmable, generic texture buffer memory controller, which will support both SGRAM and SDRAM, for accessing the local texture buffer. This memory controller will arbitrate between the host (via TAYRA's PCI interface through the on-chip communication FIFOs), and the graphics pipeline.

The texture buffer interface includes all the features of the depth buffer, bar the integrated depth test, with the addition of the following texture specific features:

- Textures can be stored in 32 bit, 16 bit or 8 bit formats to support efficient point, linear, bilinear, and trilinear texturing modes using minimum memory accesses.
- Accesses per pixel
    - Point: 1 (32 bit) or ½ (16 bit) or ¼ (8 bit) accesses
    - Linear: 2 (32 bit) or 1 (16 bit) or ½ (8 bit) accesses
    - Bilinear: 4 (32 bit) or 2 (16 bit) or 1 (8 bit) accesses
    - Trilinear: 8 (32 bit) or 4 (16 bit) or 2 (8 bit) or 1 (palletised) access
- Textures are spread over the same page in multiple devices to reduce page change overheads.
- PCI to texture memory controller high bandwidth path

5.2.3 Colour Buffer Interface

TAYRA implements a programmable generic colour buffer memory controller, which will support both VRAM and WRAM, for accessing the local colour buffer. This memory controller will arbitrate between the host (via TAYRA's PCI interface through the on-chip communication FIFOs), and the graphics pipeline.

The colour buffer WRAM interface has the following functions:

- Parallel bandwidth
    - 132 MBytes/sec at 66 MHz
- Serial bandwidth
    - 1.1 GBytes/sec
- Memory operations supported include
    - Read and write
    - Page mode read and write
    - Masked write
    - Block masked write
- Buffer clear bandwidths
    - 1.1 GBytes/sec for 1 memory chip (WRAM)
    - 8.5 GBytes/sec for 8 memory chips (WRAM)
- Supports 1 to 16 MBytes of WRAM
- Supports 1 to 8 MBytes of VRAM for low cost lower performing systems
- Global refresh controller
- Interaction with 2D Graphics chips
    - Off-chip multiplexing of 2D and 3D pixel streams
    - Shared Frame Buffer Protocol
    - Advantages gained from using off-shelf 2D chips
        - Existing SVGA drivers
        - TAYRA is compatible with standard VESA modes
- 3D update rate tuned to display refresh to avoid artefacts
- Programmable single and double buffering

5.3 Video Interface and Refresh Controller

TAYRA has a video interface which synchronises with the display refresh. This video interface supports video Vsync, Hsync, Blank inputs and outputs Dot Clock and Active Area signals.

TAYRA has a global memory refresh controller which, discounting any memory refresh accesses, refreshes every row every 17 ms.

6. Graphics Pipeline Features

TAYRA implements a large set of graphics raster algorithms in a 33 MHz pipeline. A key feature of this graphics pipeline is its communication mechanism to TAYRA's external interfaces, and pipeline modules. This communication mechanism is based around several FIFOs and an asynchronous protocol called MICE [5]. These FIFOs also allow fast loading of the Program Register Sets and the depth, texture and colour buffers.

6.1 Command and Data FIFO

TAYRA implements an input command and data FIFO (CDFIFO) to buffer input commands and data sent from the host CPU. This CDFIFO will store a number of primitives waiting to be rasterised. The CDFIFO will store sufficient primitives to cope with worst case PCI bus latency in the order of 12-15 μs.

6.2 Host to Depth, Texture and Colour Buffer Access FIFOs

TAYRA implements FIFOs between the depth, texture and colour memory controller, which provides fast access to these buffers by the host processor. This fast access is only limited by the PCI bandwidth of 132 MBytes/sec. This direct access to the depth, texture and colour buffers allows TAYRA to implement advanced graphics functions, such as stencilling and multi-pass antialiasing. Also, 3D scenes can be loaded extremely fast into the colour buffer to serve as a rendered background colour. This is very useful for video backdrops..

6.3 Depth Test

TAYRA implements eight different depth comparison tests, which provides for a very flexible Z-buffer algorithm. This depth buffer test is also used in transparency, and stencil based algorithms. TAYRA's depth test operations are:

1. If (TRUE) always write Znew
2. If (FALSE) never write Znew
3. If (Znew > Zold) write Znew
4. If (Znew >= Zold) write Znew
5. If (Znew < Zold) write Znew
6. If (Znew <= Zold) write Znew
7. If (Znew == Zold) write Znew
8. If (Znew != Zold) write Znew

TAYRA's depth test is tightly integrated in the depth memory controller allowing a single cycle read, depth compare, write operation.

6.4 Program State Registers

TAYRA implements a large register set that constitutes the 'state' of the chip. Setup operations for TAYRA involve writing values into some or all of these registers. These registers are referred to as the *program state registers*. TAYRA's program register set is associated with three main functionalities: PCI Interface, Graphics Pipeline and the three memory controllers. Two types of data are loaded into these registers: commands and data.

The program state registers can be loaded with special modes:

- Context addressing to speed up primitive loading
- DMA burst transfers with or without context addressing

6.5 Graphics Raster Pipeline

TAYRA's graphics raster pipeline implements many common rasterisation algorithms and graphics primitives :

- Scan conversion of lines, spans, triangles and rectangles
- Flat and Gouraud shading
- Specular shading
- Hidden surface elimination through depth buffering
- Texturing including colour mapping, tiling, bill boarding, mip map support, etc.
- OpenGL alpha blending including antialiasing and transparency
- Stencilling

6.5.1 Primitive Scan Conversion

TAYRA scan converts four basic geometric primitives for drawing. These are the line segment, span, triangle and rectangle. The same data path hardware is leveraged to draw variations of all four primitives.

6.5.1.1 Lines

TAYRA implements options for drawing lines with the following attributes: constant colour, colour interpolated (depth cued), constant depth, depth interpolated. TAYRA can also implement wireframe objects with the following attributes: hidden lines removed, colour interpolated.

6.5.1.2 Wide Lines

TAYRA also implements wide lines with the same features as for normal lines. This achieved by a special setup procedure which divides a wide line in two and then draws both halves sequentially. Also, antialiasing can be switched on for wide lines.

6.5.1.3 Spans

TAYRA has the option to implement horizontal and vertical spans when texture mapping with perspective correction disabled. Drawing of textured spans is common is many games applications. Spans do not need to be perspectively corrected at pixel rates, and so computational savings are made by the host during setup.

6.5.1.4 Triangles

TAYRA implements triangles which can be rasterised in many modes: wireframe, hiddenline, flat shaded, Gouraud shaded, alpha blended (antialiased and transparent) texture mapped, perspectively corrected, etc.

6.5.1.5 Rectangles

TAYRA implements rectangles which can be used in buffer clearing or background fill modes.

6.5.2 Pixel Clipping

TAYRA clips pixels to regions. Each TAYRA primitive has a bounding box defined by xmin, ymin and xmax, and ymax. Also a screen region in which TAYRA rasterises is loaded. TAYRA clips pixels to the region of the bounding box which lies inside the screen region. This eliminates the generation of invalid pixels which would lie outside the screen region. This pixel clipping can be leveraged by multiple TAYRA architectures using screen space subdivision where the host can simply send primitives which cross screen space subdivision boundaries to more than one TAYRA, saving the host from clipping.

6.5.3 Fogging

TAYRA implements OpenGL style fogging, which can also be adapted for other atmospheric effects. OpenGL GL_EXP, GL_EXP2 and GL_LINEAR fogging hardware is implemented.

6.5.4 Shading

TAYRA implements both Flat shading and Gouraud shading.

6.5.5 Hidden Surface Elimination

TAYRA implements hidden surface elimination with a Z-buffer algorithm and hardware operating on a 32 bit Z-buffer, see depth test above.

6.5.6 Perspective Correction

TAYRA implements a per-pixel true floating point perspective correction on-chip. This corrects for perspective distortion when texture mapping arbitrary projected polygons.

6.5.7 Texture Mapping

TAYRA implements a wide variety of texture mapping modes:

- Colour mapping
- Environment mapping
- Tiling
- Transparent textures
- Mip maps
- Three different mip map organisations
- Many texture colour modes

6.5.7.1 Colour Mapping

TAYRA implements mixing of texture information and shaded pixels using the OpenGL decal or modulate modes. These modes are selected on a per-triangle basis.

6.5.7.2 Environment Mapping

TAYRA uses the OpenGL decal mode to implement environment or reflection mapping. This mode is selected on a per-triangle basis.

6.5.7.3 Tiling Modes

TAYRA implements four tiling operations which can be specified independently for the two texture coordinates:

1. Wrap mode
   - Implements a bathroom tile effect
2. Invert mode
   - Alternate tiles are inverted to ensure texture continuity at tile boundaries
3. Ignore
   - Outside of the unit square a background colour is used. The background colour can be specified on a per-triangle basis. The background colour is stored in the background colour register
4. Clamp
   - The texture colour at the periphery of the texture image is extended across the surface outside of the unit square

6.5.7.4 Mip Map Filter Modes

TAYRA implements texture mapping with mip maps and provide the following point sampling and filtering operations.

1. Point Sampling (at pixel rate)
   - The user can program TAYRA to point sample texture maps for small primitives, and primitives with small slopes in Z. A single sample is taken from the texture mip map.
2. Linear Filtering (at pixel rate)
   - TAYRA has the option to linearly filter the mip map. Two samples, one from each of two adjacent mip map levels, are taken from the texture mip map. These two samples are then linearly interpolated.
3. Bilinear Filtering (at pixel rate)
   - TAYRA has the option to bilinearly filter the mip map. Four samples from a single mip map level are taken. The four samples are then bilinearly interpolated.
4. Trilinear Filtering (at half pixel rate)
   - TAYRA has the option to trilinearly filter the mip map. Four samples are taken from a single mip map level, and four more samples are taken from an adjacent level. The eight samples are then trilinearly interpolated.

6.5.7.5 Mip Map Organisation

TAYRA supports three different mip map organisations.

1. Square mip maps
   - Each mip map has dimensions (2^n x 2^n) with each mip map having half the resolution of the previous mip map. For example, a mip map with map level dimensions of 16x16, 8x8, 4x4, 2x2, 1x1 is a square mip maps
2. Linear mip maps
   - This mip map has only a single dimension, with each map level having a dimension with a power of 2. For example, a mip map with map level dimensions of 32x1, 16x1, ... 2x1, 1x1 is a linear mip map
3. Point2D mip maps
   - This mip map has maps of the form (2^n x 2^m). In this mode the filtering operations of section 6.5.7.4 are not supported. The mip map with map levels 8x8, 8x4, 8x2, 8x1, 4x8 ... 2x2, 2x1, 1x8, 1x4, 1x2, 1x1 is a Point2D mip map

The TAYRA 1.0 Program Register Set Specification defines the mip map organisation registers.

6.5.7.6 Texture Colour Modes

TAYRA implements the following texture colour formats::

1 RGBA: 32 bits per texel, channel A is used for transparency
2 RGB5551: 16 bits per texel, with 5 bits per RGB and 1 bit for A to support transparency
3 RGB565: 16 bits per texel, with 5 bits for the R and B channel, and 6 bits for the G channel.
4 RGB676: 8 bits per texel, with 6 levels of R and B and 7 levels of G
5 RGB666: 8 bits per texel, with 6 levels for each R, G and B channel
6 RGB6761: 8 bits per texel, with 6 levels of R and B and 7 levels of G, and 1 level with value 255 decoded to mean non-visible
7 RGB6661: 8 bits per texel, with 6 levels for each R, G and B channel, and 1 level with value 255 decoded to mean non-visible

6.5.8 Blending

TAYRA will implement blending algorithms for antialiasing and transparency based on the blending equations in the OpenGL specification, see the OpenGL Programming Guide [6]. For example, TAYRA implements source and destination blending factors as follows:

1. Source Blending Factors
   - GL_ZERO
   - GL_ONE
   - GL_DST_COLOR
   - GL_ONE_MINUS_DST_COLOUR
   - GL_SRC_ALPHA
   - GL_ONE_MINUS_SRC_ALPHA
   - GL_DST_ALPHA
   - GL_ONE_MINUS_DST_ALPHA
   - GL_SRC_ALPHA_SATURATE
2. Destination Blending Factor
   - GL_ZERO
   - GL_ONE
   - GL_DST_COLOR
   - GL_ONE_MINUS_DST_COLOUR
   - GL_SRC_ALPHA
   - GL_ONE_MINUS_SRC_ALPHA
   - GL_DST_ALPHA
   - GL_ONE_MINUS_DST_ALPHA

The implementation of these source and destination factors provide TAYRA with a flexible alpha blending strategy.

6.5.8.1 Antialiasing

Antialiasing is implemented in TAYRA for lines, spans and triangles using the appropriate source and destination blending factors given above, see the OpenGL Programming Guide [6].

6.5.8.2 Transparency

TAYRA implements transparency using the appropriate source and destination blending factors given above, see the OpenGL Programming Guide [6]. For example, TAYRA can implement transparency with a single pass. However, it does require that the scene is sorted by opaque (Alpha = 1.0) polygons and transparent (Alpha < 1.0) polygons before rasterisation. All opaque polygons are drawn first, with the depth buffer enabled. Then transparent polygons are drawn next, with depth buffer test enabled but depth buffer write disabled.

6.6 Pixel Colour Resolution

TAYRA will implement colour resolutions compatible with SVGA colour resolutions as follows:

- True colour
  - 32 bit 8:8:8:8 RGBA, (16 M simultaneous colours)
  - 24 bit 8:8:8 RGB, (16 M simultaneous colours)
- Hi-Colour
  - 16 bit 5:6:5 RGB, (64 K simultaneous colours)
  - 15 bit 5:5:5 RGB, (32 K simultaneous colours)

6.7 Pixel Format Conversion

TAYRA implements several pixel format conversions between the host and the texture and colour buffers. Although the texture and colour buffers are set to 15, 16, 24 and 32 bit colours, the host or any device such as an MPEG decoder can write transparently YUV 4:2:2, YUV 4:4:4, RGB 15, 16, 24 or 32 bit. For example, if the colour buffer is set to 16 bit and the host writes 32 bit, the pixel conversion module converts from 32 to 16 bit transparently. Similarly, YUV 4:2:2 would be converted to 16 bit and so on.

- Format Conversions
  - YUV 4:2:2 to 15, 16, 24, 32
  - YUV 4:4:4 to 15, 16, 24, 32
  - RGB 15 bit to 16, 24, 32
  - RGB 16 bit to 15, 24, 32
  - RGB 24 bit to 15, 16, 32
  - RGB 32 bit to 15, 16, 24

6.8 Block Move

TAYRA implements a 3D Block Move from system memory to the colour buffer. This can be performed by the host with TAYRA in PCI Target mode, or by TAYRA when in Master mode. This allows, for example, a 3D scene to be used as the background colour. This Block Move can be done at 132 MBytes/sec, limited only by the PCI bandwidth.

6.9 Display Resolutions

Table 6-2 indicates the amount of display memory needed for the TAYRA's Display and pixel colour resolutions.

7. Conclusions

TAYRA is now well into the synthesis and final verification phase. We are opening up negotiations with ES2 for fabrication in a 0.5 or 0.6 micron process. TAYRA consists of about 270K gate equivalent logic and about 18 Kbits memory for FIFOs, registers, etc. We hope to have first sample chips by 1Q97.

We are also considering an extensive range of extra features for consideration in a future TAYRA 2 version. Most notable is a setup engine and vertex level interface which is nearing design completion. The emergence of Accelerated Graphics Port (AGP) has also focused our attention for use as the interface I a future TAYRA 2 chip. This is not a big step for us because we already have a fully compliant PCI Master/Target interface.

Display Resolution	**Single Buffer Colour Display Memory Requirement (Bytes)**			
	8 Bits	**16 Bits**	**24 Bits**	**32 Bits**
320x200	64,000	128,000	192,000	256,000
640x480	307,200	614,400	921,600	1,228,800
800x600	480,000	960,000	1,440,000	1,920,000
1024x768	786,432	1,572,864	2,359,296	3,145,728
1152x864	995,328	1,990,656	2,985,984	3,981,312
1280x1024	1,310,720	2,621,440	3,932,160	5,242,880
1024x1536	1,572,864	3,145,768	4,718,592	6,291,456
1600x1200	1,920,000	3,840,000	5,760,000	7,680,000

Table 6-2 TAYRA colour buffer memory requirements

8. Acknowledgements

TAYRA is part of the ESPRIT Monograph project funded by the European Commission. We are grateful to our Monograph partners (IBM Germany, Universität Tübingen and Caption), for their contributions to this project. We extend our gratitude to Joachim Binder of IBM for his detailed contribution while at Sussex.

9. References

[1] Pineda, Juan, "*A Parallel Algorithm for Polygon Rasterization*", pp.17-20 in Computer Graphics, Volume 22, Number 4, Addison Wesley (1988).

[2] Marcus Waller, Paul Lister, *"TAYRA Scan Conversion Module"*, Technical Report Sussex/Monograph/044, Centre for VLSI and Computer Graphics, University of Sussex, BN1 9QT, October 12, 1995.

[3] Lockheed Martin, *"R3D/100 Real Time, Real 3D—Ultimate Graphics Performance for the PC"*, Brochure by Lockheed Martin.

[4] Paul Lister, *"PCI Bus Toolkit V0.1 Data Sheet"*, Centre for VLSI and Computer Graphics, University of Sussex, BN1 9QT.

[5] Simon Pearce, Paul Lister, *"Modular Interconnect Communication Environment (MICE)"*, Technical Report UOS_00100_VLSI, Centre for VLSI and Computer Graphics, University of Sussex, BN1 9QT, October '95.

[6] Jackie Neider, Tom Davis, Mason Woo, *"OpenGL Programming Guide"*, The Official Guide to Learning OpenGL, Release 1, Addison Wesley, 1993.