PS2 high-level structure

The Emotion Engine (EE)

- Memory 32mb
- EE: 128-bit Emotion Engine
- VU0/VU1: Vector Units
- FPU: Floating Point Unit
- GS: Graphic Synthesiser
- DMA: Direct memory access
- IPU: Image processing Unit

IOP: Input Output Processor
SPU2: Sound Processor

S. Ewen & L. Lemarie, “Console Yourself”
Special subprocessors

• IOP Input/Output Processor
  – Contains R3000 (provides PS1 compatibility)
  – 2 MB memory (same as PS1)
  – Controllers, memory cards, SPU2, DVD drive, USB, “Firewire”

• SPU2 Sound Processing Unit
  – 2 DSP cores, 48 channels
  – 2 MB sound memory

Emotion Engine

- 300 MHz
- MIPS III core
- Two “Vector Units”
- Graphics Interface (GIF) for talking to Graphics Synthesizer (GS)
- Image Processing Unit
  - MPEG2 decoder
  - Macroblock decoding
  - Vector quantization

Emotion Engine - high-level structure

Vector Processing Units - capabilities

• 16, 16-bit integer registers
• 32, 128-bit floating point registers
  – Split into 32 bit words (x,y,z,w)
• Four FMACs in one clock cycle
• 1 floating-point division unit
• 1 integer ALU

Vector Processing Units - roles

• VPU0: intended for “thought simulation and physical simulation”
  – Outputs to ScratchPad RAM (SPR) for use by GS for VPU1
  – 4K data/4K instruction

• VPU1: intended for graphics pipeline
  – Geometry transformation
  – Vertex lighting
  – Outputs triangles (display list) to Graphic Synthesizer
  – 16K data/16K instruction

Connection styles

Parallel Connection

Serial Connection

(c) IEEE 1999 1999 ISSCC Slide Supplement / Copyright IEEE
Vector Unit 0

Vector Unit 1

Caches and scratchpad

- Similar to old style PC L1 cache
- PS2 has small caches, as it was felt that a lot of dynamic data would not be in the cache for any length of time

From D. Carter, “Introducing PS2 to PC Programmers,” AGDC 2002
Vector Processing Units

Intro slides for UCSD “CSE 191A: Video Game Programming Seminar” pisa.ucsd.edu/cse191/www/CSE191_01.ppt
Typical vsm assembly (dual stream)

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOP</td>
<td>lq</td>
</tr>
<tr>
<td>mulax</td>
<td>VF06,7(VI02)</td>
</tr>
<tr>
<td>madday</td>
<td>VI01,VI01,0x00000001</td>
</tr>
<tr>
<td>maddaz</td>
<td>VI02,VI02,0x00000002</td>
</tr>
<tr>
<td>maddw</td>
<td>NOP</td>
</tr>
<tr>
<td>NOP</td>
<td>NOP</td>
</tr>
<tr>
<td>NOP</td>
<td>NOP</td>
</tr>
<tr>
<td>NOP</td>
<td>NOP</td>
</tr>
<tr>
<td>mulaw.xyz</td>
<td>VF01,VF00w</td>
</tr>
<tr>
<td>NOP</td>
<td>NOP</td>
</tr>
<tr>
<td>NOP</td>
<td>NOP</td>
</tr>
<tr>
<td>NOP</td>
<td>NOP</td>
</tr>
<tr>
<td>NOP</td>
<td>NOP</td>
</tr>
<tr>
<td>mulq.xyz</td>
<td>VF06,VF06,Q</td>
</tr>
<tr>
<td>NOP</td>
<td>NOP</td>
</tr>
<tr>
<td>NOP</td>
<td>NOP</td>
</tr>
<tr>
<td>NOP</td>
<td>NOP</td>
</tr>
</tbody>
</table>

Info from powerpoint presentation by H.S. Fortuna, “Video Game Programming Using The PlayStation2 Games Console,” www.ics.heacademy.ac.uk/events/presentations/91_BCSTalk.ppt
Typical VCL (single stream)

loop:
  lq         Vert, StartVert(iVertPtr)

  MatrixMultiplyVertex  Vert, fTransform, Vert

  div        q,     vf00[w], Vert[w]
  mul.xyz   Vert, Vert, q

  mula.xyz  acc, fScales, vf00[w]
  madd.xyz Vert, Vert, fScales
  ftoi4.xyz Vert, Vert

Slide from powerpoint presentation by H.S. Fortuna, “Video Game Programming Using The PlayStation2 Games Console,”
www.ics.heacademy.ac.uk/events/presentations/91_B CSTalk.ppt
Graphics Synthesizer (GS)

- Receives display list of triangles from GIF
- Rasterizes triangles into frame buffer
- Handles z-buffering, alpha blending, texture mapping
- Outputs frame buffer to video
- Two sets of drawing environments (internal contexts)
  - GS knows which instructions came from VPU0 and VPU1
  - Merges sequences

PS2 Linux

- $200-$400 on ebay

Picture from powerpoint presentation by H.S. Fortuna, “Video Game Programming Using The PlayStation2 Games Console,”
www.ics.heacademy.ac.uk/events/presentations/91_BCSTalk.ppt