Reverse Engineering Midway Zeus
Part 1

Phil Bennett

About Me

- GPU Architect
- Arcade game enthusiast
- Occasional MAME contributor

Z-Plus Hardware

- Midway's second 3D-capable platform
- Two games produced
  - Mortal Kombat 4 (1997)
  - Invasion: The Abductors (1999)

Z-Plus Emulation

- Z-Plus supported added to MAME in 2007 by Aaron Giles
- Some improvements made over the years by me
- Far from perfect
  - No lighting model
  - Inaccurate blending/fog
  - Missing polygons
  - Polygon clipping issues
  - Rasterization errors

MAME vs Hardware

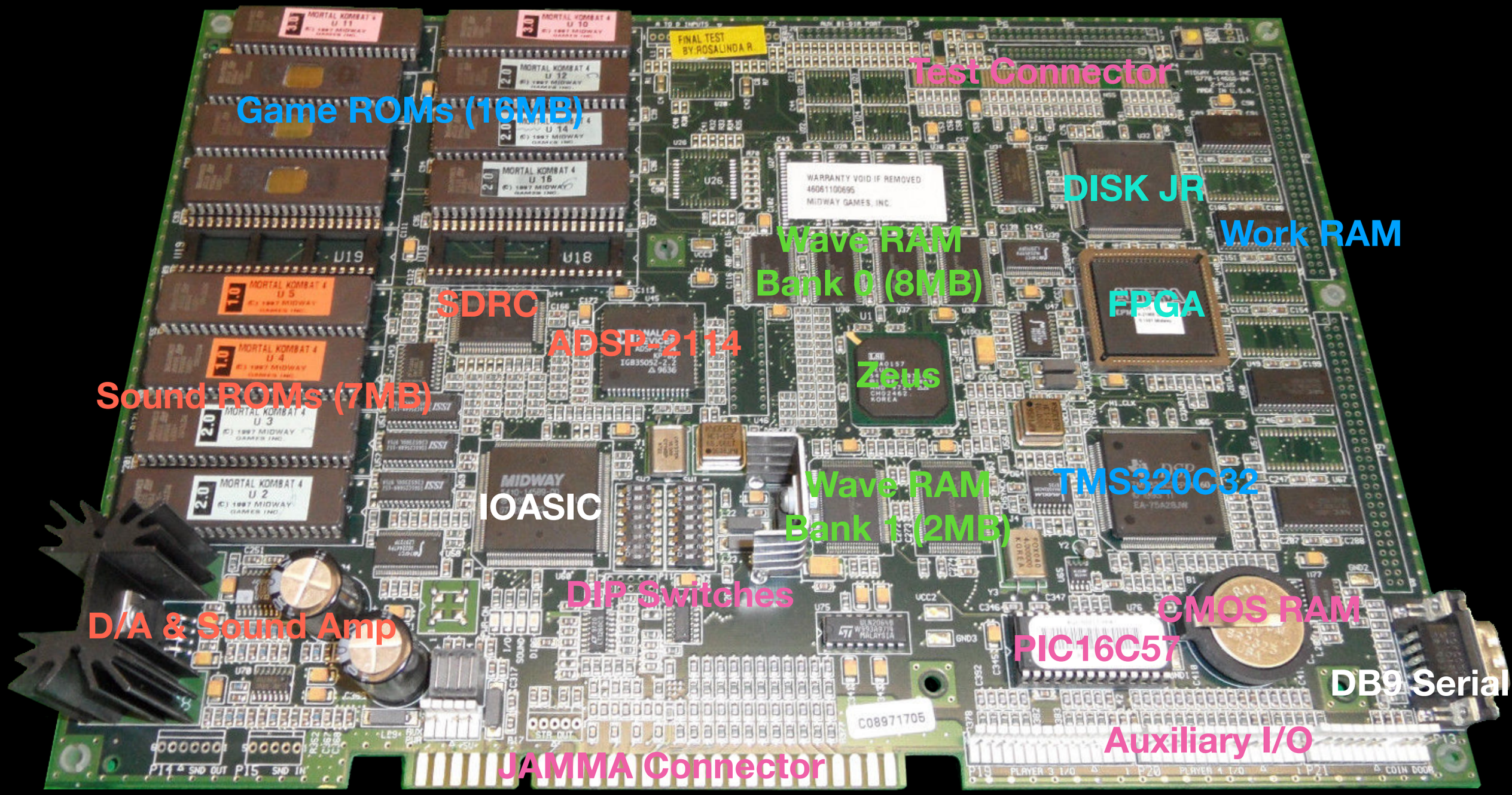
Emulation Targets

- Fully-playable (ideally at 100% speed)
- 1:1 pixel accuracy
- Reasonably accurate hardware timing

The Zeus 3D Accelerator

- Microcoded VLIW 'math machine'
- Bilinear filtering
- Gouraud shading
- Z-buffer (16-bit fixed point)
- Framebuffer R/M/W (blending)
- 33Mpixels/s
- 600,000 quads/s
- 66MHz, 250K gates, 0.5µm process

Z-Plus PCB

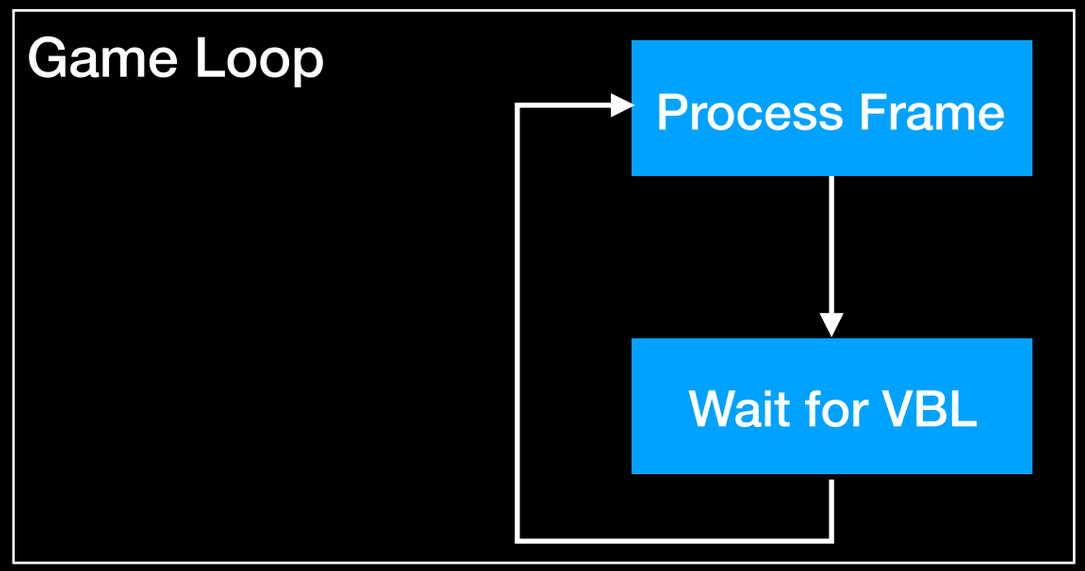

Game ROMs (16MB)
Test Connector
DISK JR
Work RAM
Wave RAM
Bank 0 (8MB)
FPGA
SDRC
ADSP-2114
Zeus
Sound ROMs (7MB)
IOASIC
Wave RAM
Bank 1 (2MB)
TMS320C32
DIP Switches
CMOS RAM
PIC16C57
D/A & Sound Amp
DB9 Serial
Auxiliary I/O
JAMMA Connector

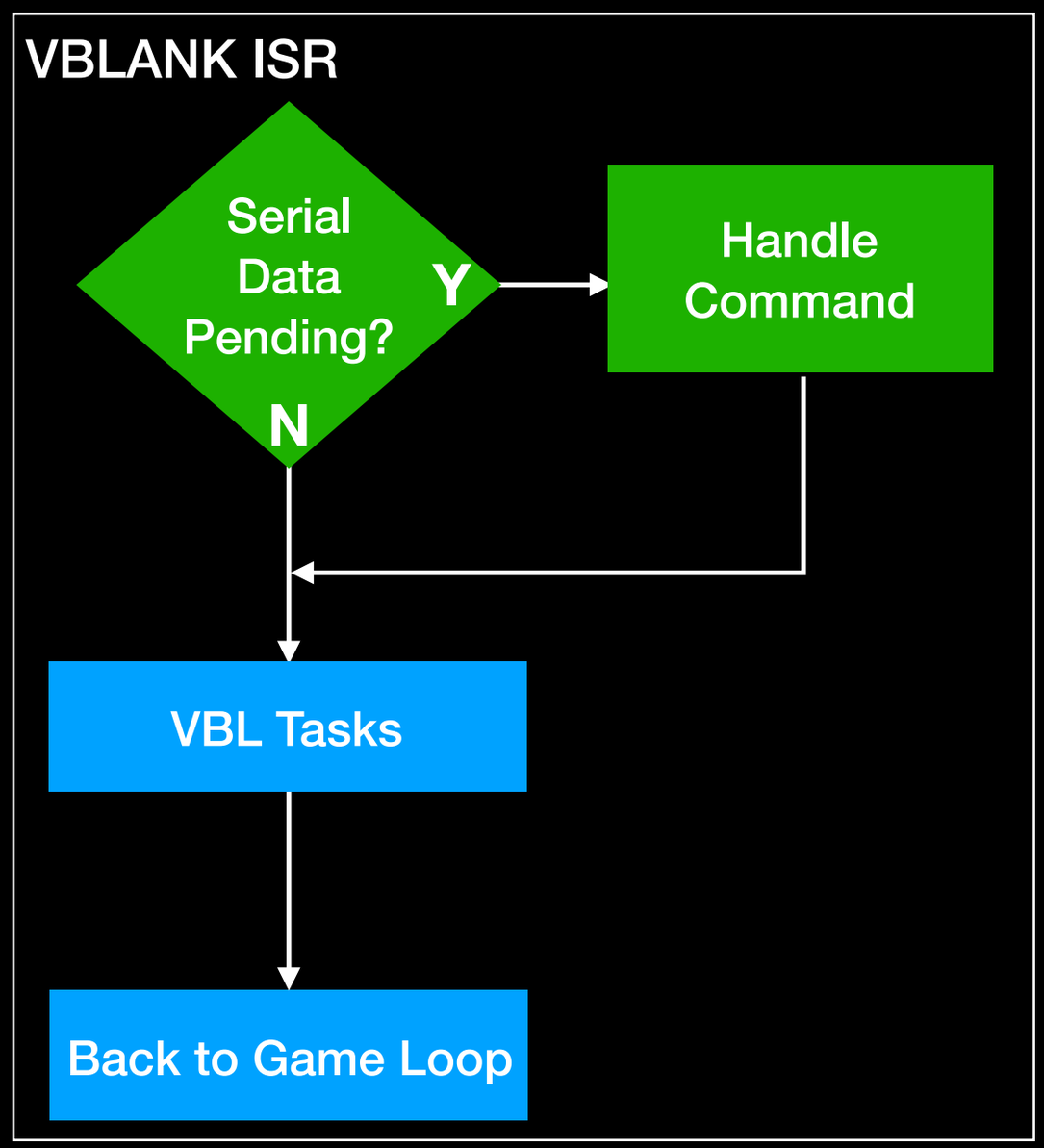
IOASIC

- Found on various Midway games
- Interface between main CPU and sound CPU
- Interface between main CPU and security PIC
- I/O interface (e.g. joystick and coin switch inputs)
- UART for linking multiple games

Plan of Attack

- Use IOASIC UART to communicate between Z-Plus and host PC
  - Linux driver for IOASIC found on HDD of Midway Skins Game
- Modify game to allow host to control game execution
  - TMS320 assembly
- Write host program to send/receive data and automate testing
  - C++/Lua
- Give it a daft name: Dr. Zeus



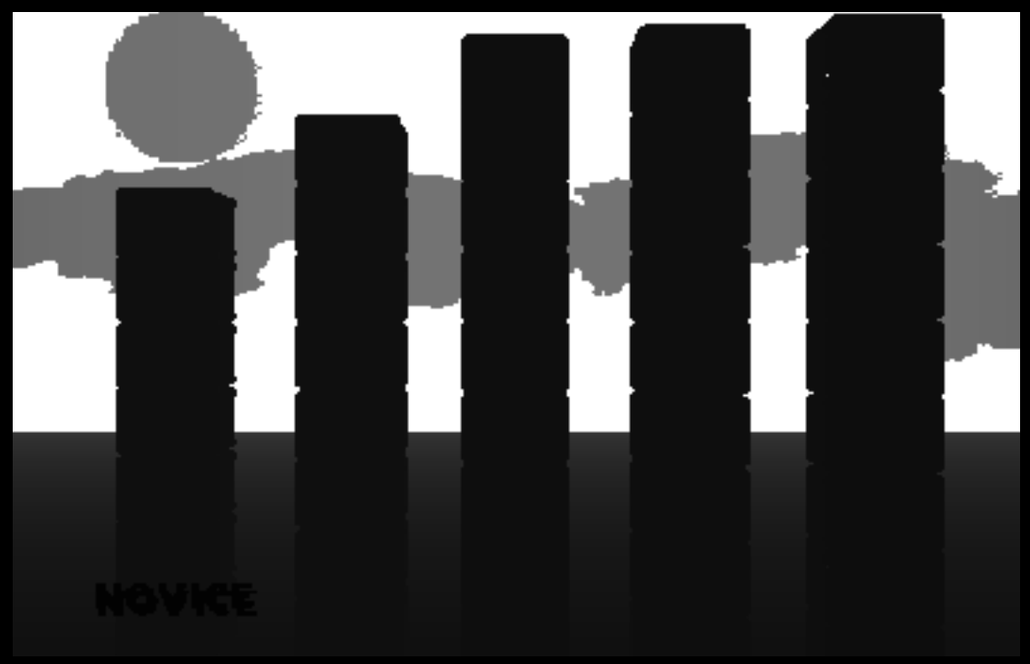



```
;=============================================================================
; uart_isr:
;=============================================================================
process_cmd:
                PUSH    R0
                PUSH    AR0
                PUSH    AR1
                PUSH    AR2

                LDI     @MONITOR_CMD_IDX, AR0
                LDI     @MONITOR_CMD_CNT, AR2

                ; Do the command
                CMPI            MONITOR_HOST_CMD_DISABLE_BREAK,AR0
                BEQ             cmd_0
                CMPI            MONITOR_HOST_CMD_ENABLE_BREAK,AR0
                BEQ             cmd_1
                CMPI            MONITOR_HOST_CMD_IDX_CONTINUE,AR0
                BEQ             cmd_2
                CMPI            MONITOR_HOST_CMD_READ_MEMORY,AR0
                BEQ             cmd_3
                CMPI            MONITOR_HOST_CMD_WRITE_MEMORY,AR0
                BEQ             cmd_4
                CMPI            MONITOR_HOST_CMD_READ_WAVERAM,AR0
                BEQ             cmd_5
                CMPI            MONITOR_HOST_CMD_WRITE_WAVERAM,AR0
                BEQ             cmd_6
                CMPI            MONITOR_HOST_CMD_WRITE_FIFO,AR0
                BEQ             cmd_7
                CMPI            MONITOR_HOST_CMD_TEST,AR0
                BEQ             cmd_8

                LDI             @MONITOR_STATE,R0                       ; ERROR
                ANDN            MONITOR_STATE_ERROR_MASK,R0
                OR              MONITOR_STATE_ERROR_BAD_CMD,R0
                STI             R0,@MONITOR_STATE
                LDI             0,R0                                    ; No reply words

                BR              cmd_exit

cmd_0: ; Disable halt on VBL
                LDI             @MONITOR_STATE,R0
                ANDN            MONITOR_STATE_BREAK_VBL | MONITOR_STATE_HALTED,R0
                STI             R0,@MONITOR_STATE
                LDI             00h,R0
                BU              cmd_exit

cmd_1: ; Enable halt on VBL
                LDI             @MONITOR_STATE,R0
                OR              MONITOR_STATE_BREAK_VBL,R0
                STI             R0,@MONITOR_STATE
                LDI             00h,R0
                BU              cmd_exit

cmd_2: ; Continue
                LDI             @MONITOR_STATE,R0
                ANDN            MONITOR_STATE_HALTED,R0
                STI             R0,@MONITOR_STATE
                LDI             00h,R0                                  ; Reply count of 0
                BU              cmd_exit

cmd_3: ; Read memory
                LDI             @MONITOR_CMD_DATAPTR+0,AR0              ; Source Address
                LDI             @MONITOR_CMD_DATAPTR+1,AR2              ; Count
                LDI             MONITOR_REPLY_DATAPTR,AR1               ; Temp dest
                CALL            memcpy
                LDI             AR2,R0                                  ; Reply count
                BU              cmd_exit

cmd_4: ; Write memory
                LDI             @MONITOR_CMD_DATAPTR+0,AR1              ; Destination Address
                LDI             MONITOR_CMD_DATAPTR+2,AR0               ; Source pointer
                LDI             @MONITOR_CMD_DATAPTR+1,AR2              ; Count
                CALL            memcpy
                LDI             0,R0                                    ; Reply count of 0
                BU              cmd_exit
```

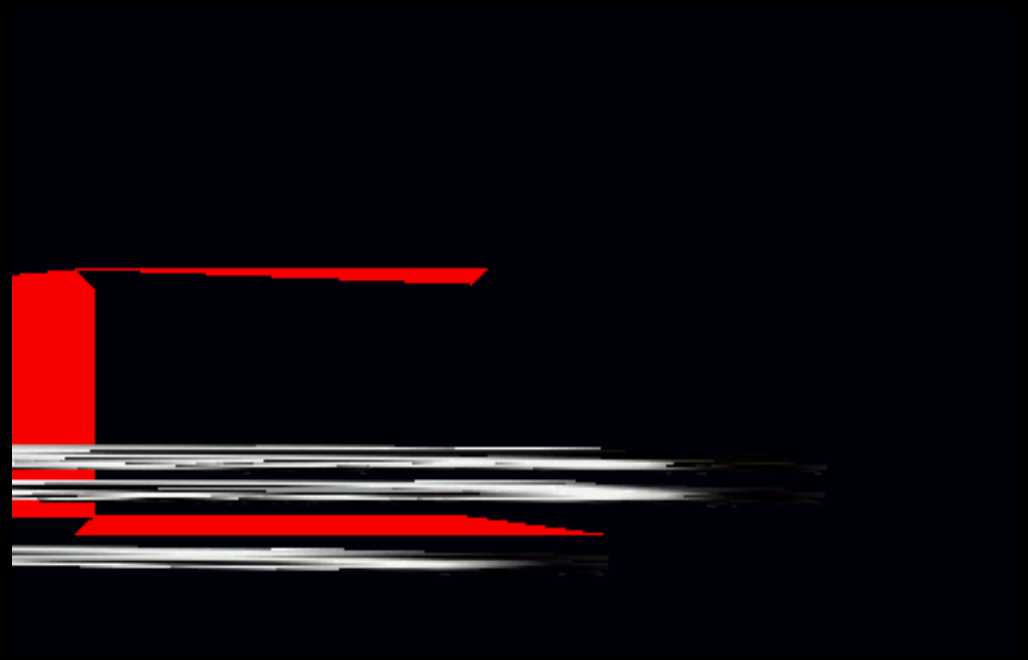
Dr. Zeus Commands

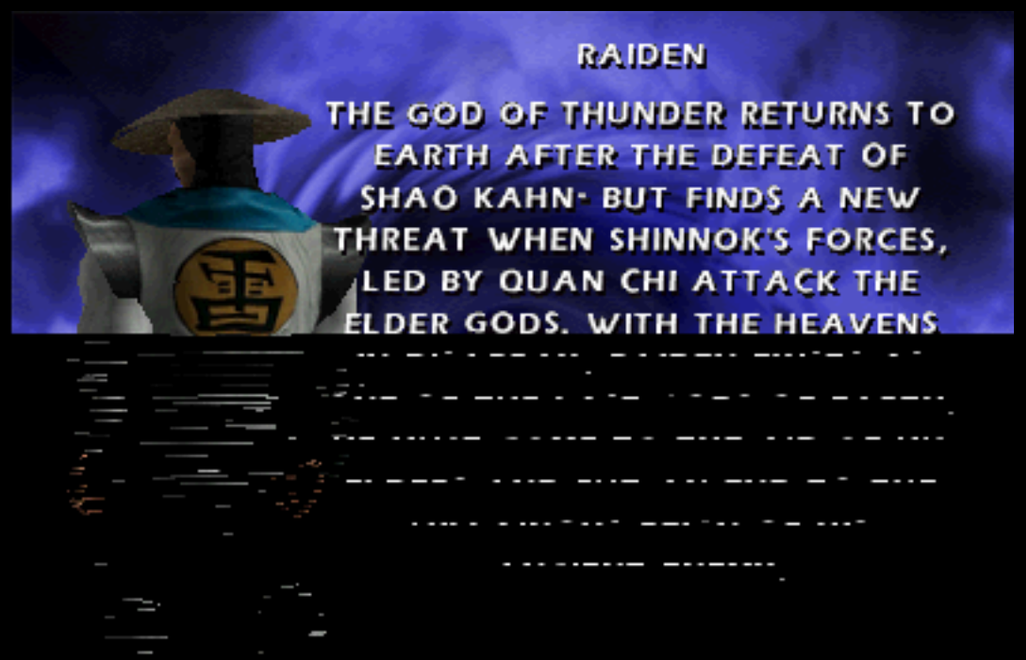
- Halt game execution at next VBL
- Single-step frame
- Send Zeus FIFO command
- Write math machine microcode
- Read/write Zeus registers
- Read/write work RAM
  - Game code executes from RAM, allowing on-the-fly patching
- Read/write wave/video RAM
  - Dump colour and depth buffers

Colo(u)r and Z-Buffer Captures



What if I Change...





Part 2

- Writing effective, directed hardware tests
- The Math Machine instruction set
- The Zeus 3D engine
- Zeus II