

Simulator

Author

Contents

Project
Review

Project Target
Review
Expected
Progress

Project
Introduction

Module View
File View

Module
Analysis

Process Module
CPU Module
Memory Module
Cache and
Register Module
Pipeline Module

Project
Bottleneck

Future Work

Unicore Simulator Interim Report

Li Chunqi Peng Zhuo
Wang Kan Hua Liansheng

December 3, 2010

Simulator

Author

Contents

Project
Review

Project Target
Review
Expected
Progress

Project
Introduction

Module View
File View

Module
Analysis

Process Module
CPU Module
Memory Module
Cache and
Register Module
Pipeline Module

Project
Bottleneck

Future Work

Contents

- 1 Project Review
 - Project Target Review
 - Expected Progress
- 2 Project Introduction
 - Module View
 - File View
- 3 Module Analysis
 - Process Module
 - CPU Module
 - Memory Module
 - Cache and Register Module
 - Pipeline Module
- 4 Project Bottleneck
- 5 Future Work

Simulator

Author

Contents

**Project
Review**

Project Target
Review
Expected
Progress

**Project
Introduction**

Module View
File View

**Module
Analysis**

Process Module
CPU Module
Memory Module
Cache and
Register Module
Pipeline Module

**Project
Bottleneck**

Future Work

Project Review

Project Target Review

Simulator

Author

Contents

Project
Review

Project Target
Review

Expected
Progress

Project
Introduction

Module View
File View

Module
Analysis

Process Module
CPU Module
Memory Module
Cache and
Register Module
Pipeline Module

Project
Bottleneck

Future Work

- Realize CPU controller and datapath with **five** level pipeline
- Realize at least one level cache, **Havard architecture**.
- Realize dynamic memory management.
- Support of some system library functions.
- Debugger utils and Performance Analysis utils support.

Expected Progress

Simulator

Author

Contents

Project
Review

Project Target
Review
Expected
Progress

Project
Introduction

Module View
File View

Module
Analysis

Process Module
CPU Module
Memory Module
Cache and
Register Module
Pipeline Module

Project
Bottleneck

Future Work

- 1-2 weeks ELF parser module, Register heap module, Memory module. (Finished in 1st week)
- 3-4 weeks CPU module. (Finished in 4th week)
- 5-6 weeks Cache module(Finished in 3rd week), Loader module. (Finished in 2nd week)
- 7-8 weeks Debugger module and some latter works.(To be finished in 1-2 week)

More infomation about our project progress, see our svn:

<http://code.google.com/p/minic/wiki/MINICintroduction?tm=6>

Simulator

Author

Contents

Project
Review

Project Target
Review
Expected
Progress

**Project
Introduction**

Module View
File View

Module
Analysis

Process Module
CPU Module
Memory Module
Cache and
Register Module
Pipeline Module

Project
Bottleneck

Future Work

Project Introduction

Module View

Simulator

Author

Contents

Project
Review

Project Target
Review
Expected
Progress

Project
Introduction

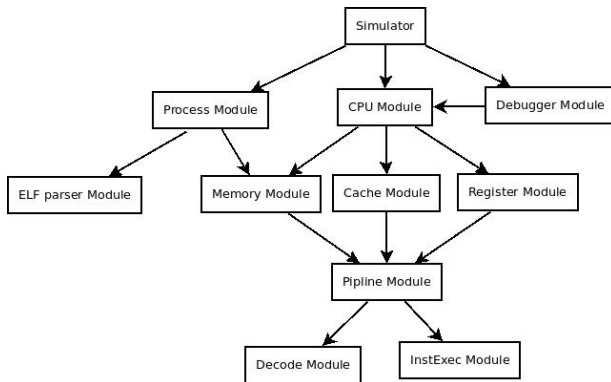
Module View
File View

Module
Analysis

Process Module
CPU Module
Memory Module
Cache and
Register Module
Pipeline Module

Project
Bottleneck

Future Work



File View

Simulator

Author

Contents

Project

Review

Project Target

Review

Expected

Progress

Project

Introduction

Module View

File View

Module

Analysis

Process Module

CPU Module

Memory Module

Cache and

Register Module

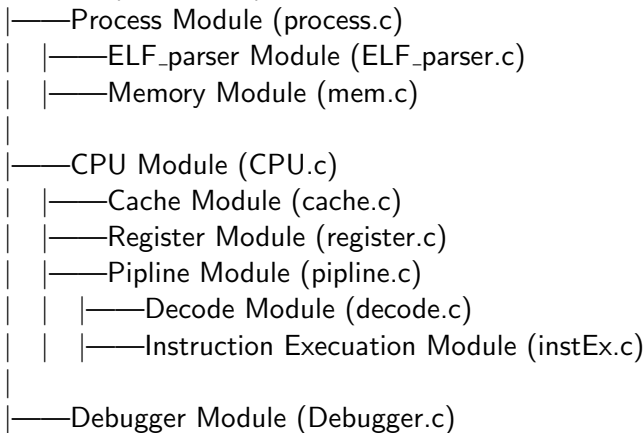
Pipeline Module

Project

Bottleneck

Future Work

Simulator (simulator.c)



Simulator

Author

Contents

Project
Review

Project Target
Review
Expected
Progress

Project
Introduction

Module View
File View

**Module
Analysis**

Process Module
CPU Module
Memory Module
Cache and
Register Module
Pipeline Module

Project
Bottleneck

Future Work

Module Analysis

Process Module

Simulator

Author

Contents

Project
Review

Project Target
Review
Expected
Progress

Project
Introduction

Module View
File View

Module
Analysis

Process Module
CPU Module
Memory Module
Cache and
Register Module
Pipeline Module

Project
Bottleneck

Future Work

Struct of Process:

```
typedef struct {  
    int status; //Process status  
    uint32_t entry; //The entry of a program  
    PROC_STACK* stack; //Process stack  
    PROC_MEM* mem; //Process memory  
} PROCESS;
```

Process is a basic module handles a copy of a progrss in the memory.

CPU Module

Simulator

Author

Contents

Project
Review

Project Target
Review
Expected
Progress

Project
Introduction

Module View
File View

Module
Analysis

Process Module
CPU Module
Memory Module
Cache and
Register Module
Pipeline Module

Project
Bottleneck

Future Work

CPU Module is the most important module, it lays on the top level and handles add the execution of a progress.

Struct of CPU:

```
typedef struct {  
    int cpu_id; //CPU id  
    int mode; //CPU mode, normal or trap  
    REGISTERS* regs; //Register heap  
    CACHE *i_cache , *d_cache;  
    PIPELINE * pipeline; //CPU pipeline  
    PROCESS* proc; //Process running on CPU now  
    CPU_info* cpu_info; //Information of CPU  
                                //from starting  
}CPU_d;
```

CPU Module(cont.)

Simulator

Author

Contents

Project
Review

Project Target
Review
Expected
Progress

Project
Introduction

Module View
File View

Module
Analysis

Process Module
CPU Module
Memory Module
Cache and
Register Module
Pipeline Module

Project
Bottleneck

Future Work

Struct of CPU_info:

```
typedef struct{  
    int   cycles_total; //total cycles of cpu  
    int   cycles_work; //work cycles of cpu  
    int   bubbles; //bubbles of pipline  
    int   rd_mem_times; //times of read memory  
    int   wr_mem_times; //times of write memory  
    int   cache_visit; //times of cache visit  
    int   cache_miss; //times of cache miss  
} CPU_info;
```

Memory Module

Simulator

Author

Contents

Project
Review

Project Target
Review
Expected
Progress

Project
Introduction

Module View
File View

Module
Analysis

Process Module
CPU Module
Memory Module
Cache and
Register Module
Pipeline Module

Project
Bottleneck

Future Work

Struct of Process Memory Management:

```
typedef struct {  
    unsigned int vaddr_offset;  
    unsigned int size;  
    uint8_t *base;  
    int flag;  
}PROC_SEGMENT;  
  
typedef struct {  
    unsigned int seg_num;  
    PROC_SEGMENT * segments;  
}PROC_MEM;  
  
typedef PROC_SEGMENT PROC_STACK;
```

Cache and Register Module

Simulator

Author

Contents

Project

Review

Project Target

Review

Expected

Progress

Project

Introduction

Module View

File View

Module

Analysis

Process Module

CPU Module

Memory Module

Cache and

Register Module

Pipeline Module

Project

Bottleneck

Future Work

Struct of Cache and Register Module:

```
typedef struct {
    int block_num;
    int sign_bits_num;
    PROC_MEM* mem;
    int valid [CACHE_SIZE/BLOCK_SIZE];
    uint8_t data [CACHE_SIZE/BLOCK_SIZE] [ BLOCK_SIZE];
    uint32_t mark [CACHE_SIZE/_BLOCK_SIZE];
}CACHE;
```

```
typedef struct {
    int32_t r [32];
    int32_t flag;
}REGISTERS;
```

Pipeline Module

Simulator

Author

Contents

Project

Review

Project Target

Review

Expected

Progress

Project

Introduction

Module View

File View

Module

Analysis

Process Module

CPU Module

Memory Module

Cache and

Register Module

Pipeline Module

Project

Bottleneck

Future Work

Struct of Pipeline:

```
typedef struct {  
    int block; //1 means pipeline block, 0 mean the  
    PIPELINE_DATA* pipeline_data [PIPELINE_LEVEL];  
    int using_regs [31];  
    PROC_STACK* stack;  
    REGISTERS* regs;  
    CACHE *i_cache , *d_cache;  
} PIPELINE;
```

Total Lines

Simulator

Author

Contents

Project

Review

Project Target

Review

Expected

Progress

Project

Introduction

Module View

File View

Module

Analysis

Process Module

CPU Module

Memory Module

Cache and

Register Module

Pipline Module

Project

Bottleneck

Future Work

Project has 1987 lines until now.(Exclude blank lines)

```
love@love-laptop: ~/minic/simulator
文件(F) 编辑(E) 查看(V) 终端(T) 帮助(H)
love@love-laptop:~/minic/simulator$ python count_line.py .
instEx.h : 6 lines.
pipeline.c : 171 lines.
pipeline.h : 48 lines.
ELF_parser.h : 13 lines.
test.c : 10 lines.
cache.c : 141 lines.
process.c : 54 lines.
debugger.h : 20 lines.
cache.h : 28 lines.
instEx.c : 654 lines.
decode.c : 206 lines.
register.h : 23 lines.
simulator.c : 79 lines.
process.h : 19 lines.
register.c : 73 lines.
mem.c : 123 lines.
debugger.c : 32 lines.
CPU.h : 25 lines.
mem.h : 32 lines.
inst.h : 45 lines.
decode.h : 6 lines.
CPU.c : 53 lines.
ELF_parser.c : 126 lines.
----Total : 1987 lines.----
love@love-laptop:~/minic/simulators
```


Simulator

Author

Contents

Project
Review

Project Target
Review
Expected
Progress

Project
Introduction

Module View
File View

Module
Analysis

Process Module
CPU Module
Memory Module
Cache and
Register Module
Pipeline Module

Project
Bottleneck

Future Work

Summary Before Mid-term Check

Project Bottleneck

Simulator

Author

Contents

Project
Review

Project Target
Review
Expected
Progress

Project
Introduction

Module View
File View

Module
Analysis

Process Module
CPU Module
Memory Module
Cache and
Register Module
Pipeline Module

Project
Bottleneck

Future Work

Some problems when do the project

- ① Keeping each module independent is important but difficult.
- ② System library function handling is yet left to be realized.
- ③ Some instructions are not realized because of lacking manual.
- ④ Instruction execution module is hard to test and verify, because constructing test set is sometimes ambiguous.
- ⑤ Too many modules leads to high maintenance cost.
- ⑥ Some potential bugs need to be resolved.

Future Work

Simulator

Author

Contents

Project
Review

Project Target
Review
Expected
Progress

Project
Introduction

Module View
File View

Module
Analysis

Process Module
CPU Module
Memory Module
Cache and
Register Module
Pipeline Module

Project
Bottleneck

Future Work

Future work in next 1-2 weeks:

- Complete Debugger module and high level console.
- Finish verification outline.
- Construct high-level test set(generate by c files) and low-level test set(generate by assembler files).
- Discuss and solve the problem of system library function call.
- A complete program compiled by uncore32-linux-gcc(the given compiler from lab) can be run completely in out simulator.

Simulator

Author

Contents

Project
Review

Project Target
Review
Expected
Progress

Project
Introduction

Module View
File View

Module
Analysis

Process Module
CPU Module
Memory Module
Cache and
Register Module
Pipeline Module

Project
Bottleneck

Future Work

Q&A