

LAB 02: Introduction to MIPS Assembly Programming

Saleh AlSaleh
salehs@kfupm.edu.sa

King Fahd University of Petroleum and Minerals
College of Computing and Mathematics
Computer Engineering Department

COE301: Computer Architecture
Term 222

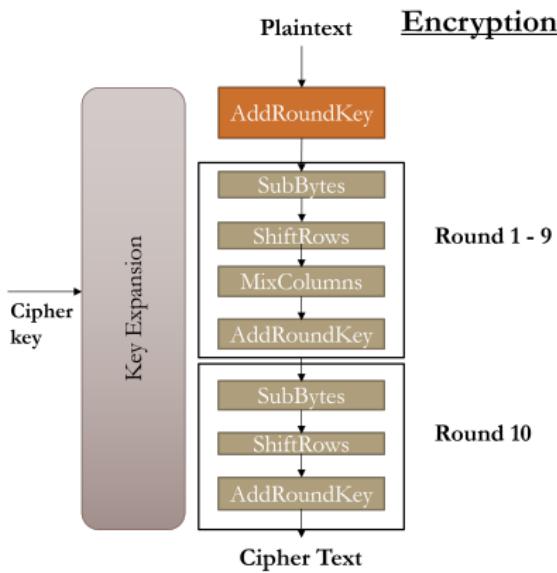
Agenda

- ➊ Prev. Side Project
- ➋ Assembly Template
- ➌ MIPS Instr. Formats
- ➍ MIPS Registers & System Calls
- ➎ Live Examples
- ➏ Tasks

Previous Side Project: AES Overview

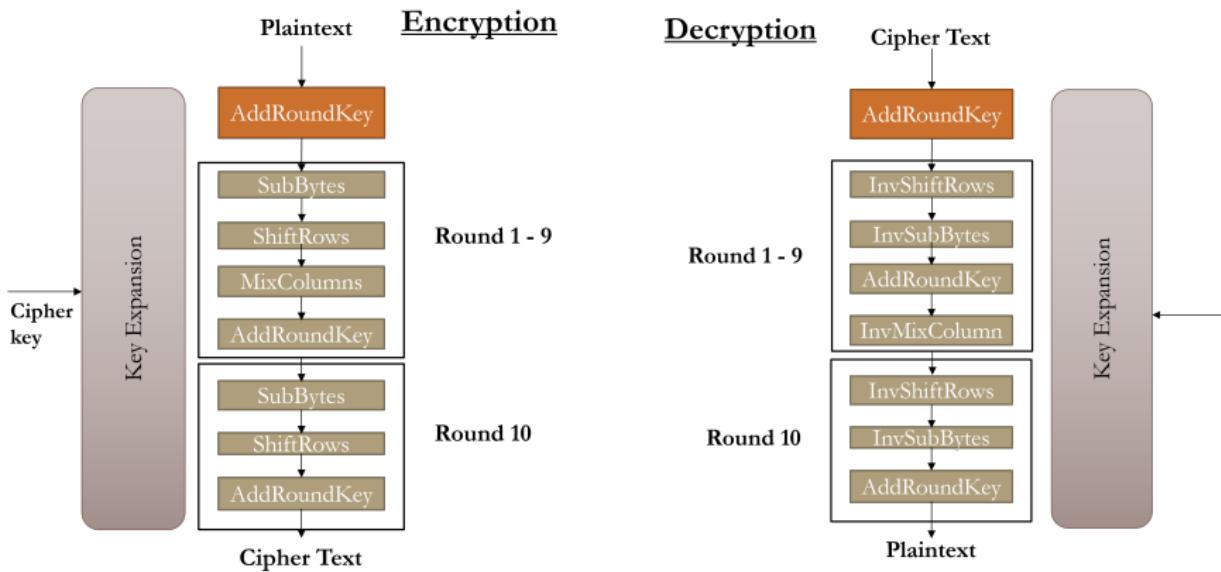
- Advanced Encryption Standard (AES) or Rijndael developed by Vincent Rijmen and Joan Daemesn is a symmetric key encryption and decryption algorithm.
- It was first published in 1998, and standardized in 2001 by U.S. National Institute of Standards and Technology (NIST).
- AES is widely used to secure connection between clients and servers.
- It has fixed data block size (128 bits) but different key lengths (128, 192, and 256 bits).

Previous Side Project: AES Algorithm



AES Encryption Algorithm

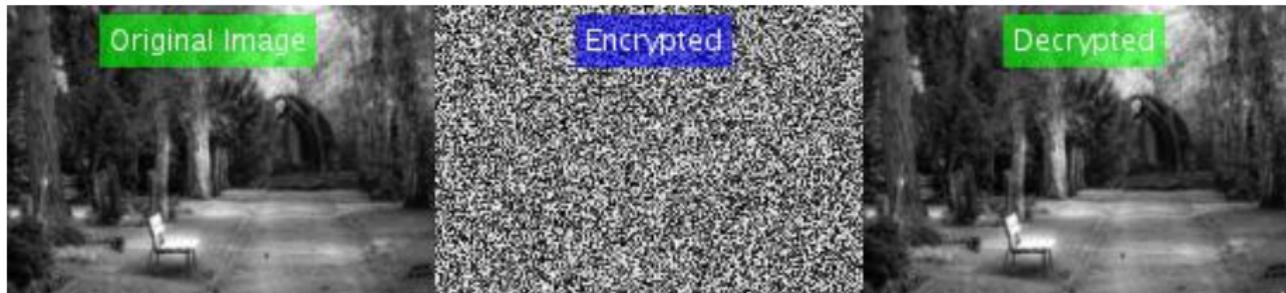
Previous Side Project: AES Algorithm



AES Encryption Algorithm

AES Decryption Algorithm

Previous Side Project: MIPS Implementation



AES MIPS Implementation and Example

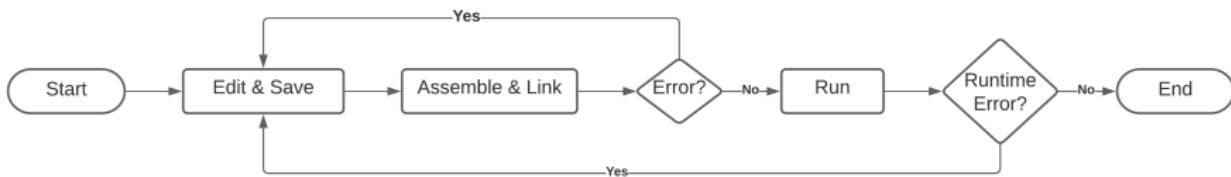
MIPS Assembly Language Program Template

```
# Title:  
# Author:  
# Date:  
# Description:  
# Input:  
# Output:  
##### Data segment #####  
.data  
...  
##### Code segment #####  
.text  
.globl main  
main:           # main function entry  
...  
li $v0, 10  
syscall         # system call to exit program
```

MIPS Assembly Program Language

- Comments start with '#'
- Directives start with '.'
- Instructions are usually written in lower case; however, you can write them in uppercase as well.
- Registers name or number starts with '\$'.

Edit-Assemble-Link-Run Cycle



MIPS Instr. Formats

- R-Type Format
 - Requires two register operands
 - e.g. add \$t0, \$t1, \$t2

op^6	rs^5	rt^5	rd^5	sa^5	$func^6$
--------	--------	--------	--------	--------	----------

R-Type Instruction Format

MIPS Instr. Formats

- R-Type Format
 - Requires two register operands
 - e.g. add \$t0, \$t1, \$t2
- I-Type Format
 - Requires two operands: a register and 16-bit immediate value
 - e.g. addi \$t0, \$t1, 301

op^6	rs^5	rt^5	rd^5	sa^5	$func^6$
--------	--------	--------	--------	--------	----------

R-Type Instruction Format

op^6	rs^5	rt^5	$immediate^{16}$
--------	--------	--------	------------------

I-Type Instruction Format

MIPS Instr. Formats

- R-Type Format
 - Requires two register operands
 - e.g. add \$t0, \$t1, \$t2
- I-Type Format
 - Requires two operands: a register and 16-bit immediate value
 - e.g. addi \$t0, \$t1, 301
- J-Type Format
 - Used for jump instructions with 26-bit immediate value
 - e.g. j loop

op^6	rs^5	rt^5	rd^5	sa^5	$func^6$
--------	--------	--------	--------	--------	----------

R-Type Instruction Format

op^6	rs^5	rt^5	$immediate^{16}$
--------	--------	--------	------------------

I-Type Instruction Format

op^6	$immediate^{26}$
--------	------------------

J-Type Instruction Format

MIPS General Purpose Registers

Register Name	Register No.	Register Usage
\$zero	\$0	Always zero, forced by hardware
\$at	\$1	Assembler Temporary register, reserved for assembler use
\$v0 - \$v1	\$2 - \$3	Results of a function
\$a0 - \$a3	\$4 - \$7	Arguments of a function
\$t0 - \$t7	\$8 - \$15	Registers for storing temporary values
\$s0 - \$s7	\$16 - \$23	Registers that should be saved across function calls
\$t8 - \$t9	\$24 - \$25	Registers for storing more temporary values
\$k0 - \$k1	\$26 - \$27	Registers reserved for the OS kernel use
\$gp	\$28	Global Pointer register that points to global data
\$sp	\$29	Stack Pointer register that points to top of stack
\$fp	\$30	Frame Pointer register that points to stack frame
\$ra	\$31	Return Address register used to return from a function call

System Calls

System calls provide system services, mainly for input and output, are available for use by your MIPS program.

Partial List of System Calls

Service	code in \$v0	Arguments	Results
Print Integer	1	\$a0 = integer to print	
Print String	4	\$a0 = address of null-terminated string	
Read Integer	5		\$v0 = integer read
Read String	8	\$a0 = address of input buffer \$a1 = maximum number of characters	
Exit Program	10		Terminate Program
Print Character	11	\$a0 = character to print	
Read Character	12		

Live Examples

Task #1

Write a MIPS program where you ask the user to enter 3 integers **a**, **b**, and **c**. Then, calculate and print the value of **z** based on the following equation.

$$z = (6a - 4b) - (3c - 20)$$

Sample Run

```
Enter a: 4
Enter b: 6
Enter c: 2
z = 14
```

Task #2

Write a MIPS program where you prompt the user for his **name**. Then, print the following message

"Welcome to COE301, <name>"

Assume the maximum length for a name is 20 characters.

Sample Run

Enter your name: Khalid

Welcome to COE301, Khalid