# **ECE366 Project 1**

Yihua PU

Program I - "6^P mod 17"

Code:

```
# Project 1 Calculate the result of "6^P mod 17"
1
   # Author: Yihua PU
2
3
   .data
4
  P: .word 1005
5
6 R: .word -1
                       # the result
7
8 even_result: .word 1,2,4,8,16,15,13,9
9 odd_result: .word 6,12,7,14,11,5,10,3
10 # the steps and ideas about how to
  # caculate these numbers are on my scratch paper attached below.
11
12
13 .text
           lw $8, 0x2000($0)
                             # load the P into register
14
                              # get the lowest bit of P
           andi $9,$8,1
15
           bne $9,$0,odd
                              # P is even or odd?
16
17 even:
                            # "6^{P} mod 17" have 16-number-cycle. So P = P mod 16
           andi $8,$8,0xF
18
           sll $8,$8,1
                              # relative word address = P*2
19
           addi $10,$0,0x2008 # the address of even_result array
20
           add $10,$10,$8
                              # the address of answer
21
           j then
22
23 odd:
           andi $8,$8,0xF  # "6^P mod 17" have 16-number-cycle. So P = P mod 16
24
           addi $8,$8,-1
                              # relative word address = (P-1)*2
25
           sll $8,$8,1
26
           addi $10,$0,0x2028 # the address of odd_result array
27
           add $10,$10,$8 # the address of answer
28
29 then:
           lw $11,($10)
30
           sw $11,0x2004($0)
31
32
```

#### If you wanna run and test it:

```
# Project 1 Calculate the result of "6^P mod 17"
# Author: Yihua PU
.data
P: .word 1005
R: .word -1
                    # the result
even_result: .word 1,2,4,8,16,15,13,9
odd result: .word 6,12,7,14,11,5,10,3
# the steps and ideas about how to
# caculate these numbers are on my scratch paper attached below.
.text
      lw \$8, 0\times2000(\$0) # load the P into register
      andi $9,$8,1  # get the lowest bit of P bne $9,$0,odd  # P is even or odd ?
even:
      andi \$8,\$8,0xF # "6^P mod 17" have 16-number-cycle. So P = P mod 16
      sll $8,$8,1
                          # relative word address = P*2
      addi $10,$0,0x2008 # the address of even_result array
                                # the address of answer
      add $10,$10,$8
      j then
odd:
      andi \$8,\$8,0xF # "6^P mod 17" have 16-number-cycle. So P = P mod 16
      addi $8,$8,-1
                          # relative word address = (P-1)*2
      sll $8,$8,1
      addi $10,$0,0x2028  # the address of odd_result array
      add $10,$10,$8  # the address of answer
then:
      lw $11,($10)
      sw $11,0x2004($0)
```

**Text Segment** 

Data Segment

Value (+10)

12

Value (+1

Value (+c)

6

### Questions:

1) Level 2 with low DIC

2)

Address

Value (+0)

0x00002000 0x00002020

0×00002040

Value (+4)

13

10

P = 8:

Bkpt	Address Code	Basic	Source				
	0x00000000 0x8c082000	lw \$8,8192(\$0)	14:	lw \$8, 0x2000(\$0)	# load the P into re	gister	
	0x00000004 0x31090001	andi \$9,\$8,1	15:	andi \$9,\$8,1	# get the lowest bi	t of P	
	0×00000008 0×15200005	bne \$9,\$0,5	16:	bne \$9,\$0,odd	# P is even or odd	?	
	0x0000000c 0x3108000f	andi \$8,\$8,15	18:	andi \$8,\$8,0xF	# "6^P mod 17" have	16-number-cycle. S	$o P = P \mod 16$
	0x00000010 0x00084040	sll \$8,\$8,1	19:	sll \$8,\$8,1	# relative word addr	ess = P*2	
	0x00000014 0x200a2008	addi \$10,\$0,8200	20:	addi \$10,\$0,0x2008	# the address of eve	n_result array	
	0x00000018 0x01485020	add \$10,\$10,\$8	21:	add \$10,\$10,\$8	# the address of ans	wer	
	0x0000001c 0x0800000d		22:	j then			
	0x00000020 0x3108000f	andi \$8,\$8,15	24:	andi \$8,\$8,0xF	# "6^P mod 17" have	•	$o P = P \mod 16$
	0x00000024 0x2108ffff		25:	addi \$8,\$8,−1	# relative word addr	ess = (P-1)*2	
	0x00000028 0x00084040		26:	sll \$8,\$8,1			
	0x0000002c 0x200a2028		27:	addi \$10,\$0,0x2028	# the address of odd		
	0x00000030 0x01485020		28:	add <b>\$10,\$10,</b> \$8	# the address of ans	wer	
	0x00000034 0x8d4b0000		30:	lw \$11,(\$10)			
	0x00000038 0xac0b2004	sw \$11,8196(\$0)	31:	sw \$11,0x2004(\$0)			
					Data Commont		
					Data Segment		
			<b>∟4</b> )	Value (+8)	Value (+c)	Value (+10)	Value (+14)
Addres							
Addres	0×00002000	8		16	1	2	4
Addres	0×00002000 0×00002020	8 13		L6 9	6 1	2	7
Addres	0×00002000	8		L6 9	1	2	
Addres	0×00002000 0×00002020	8 13		L6 9	6 1	2	7
Addres	0×00002000 0×00002020	8 13		L6 9	6 1	2	7
Addres	0×00002000 0×00002020 0×00002040	8 13		L6 9	6 1	2 2 0	7
	0×00002000 0×00002020 0×00002040	8 13		L6 9	1 6 1	2 2 0	7
	0×00002000 0×00002020 0×00002040	8 13 10 Basic	1	L6 9	1 6 1	2 2 0	7
	0x00002000 0x00002020 0x00002040	8 13 10 Basic 0 lw \$8,8192(\$0)	Source	1.6 9 3	1 6 1 0 Text Segmen	2 2 0 t	7
Bkpt	0x00002000 0x00002020 0x00002040 Address Code 0x00000000 0x8c08200	8 13 10 Basic 0 lw \$8,8192(\$0) 1 andi \$9,\$8,1	Source 14:	lw \$8, 0x2000(\$0)	Text Segmen # load the P into	2 2 0 t register bit of P	7
Bkpt	0x00002000 0x00002020 0x00002040 Address Code 0x00000000 0x8c08200 0x00000004 0x3109000	8 13 10 Basic 0 lw \$8,8192(\$0) 1 andi \$9,\$8,1 5 bne \$9,\$0,5	Source	lw \$8, 0x2000(\$0) andi \$9,\$8,1	Text Segmen  # load the P into # get the lowest	t register bit of P dd ?	7 0
Bkpt	0x00002000 0x00002020 0x00002040  Address Code 0x00000000 0x8c08200 0x00000004 0x3109000 0x00000008 0x1520000	8 13 10 Basic 0 lw \$8,8192(\$0) 1 andi \$9,\$8,1 5 bne \$9,\$0,5 f andi \$8,\$8,15	Source	lw \$8, 0x2000(\$0) andi \$9,\$8,1 bne \$9,\$0,odd	Text Segmen  # load the P into # get the lowest # P is even or or	t register bit of P dd ? ve 16-number-cycle	7 0
Bkpt	0x00002000 0x00002020 0x00002040  Address Code 0x00000000 0x8c08200 0x00000004 0x3109000 0x00000008 0x1520000 0x0000000c 0x3108000	8 13 10 Basic 0 lw \$8,8192(\$0) 1 andi \$9,\$8,1 5 bne \$9,\$0,5 f andi \$8,\$8,15 0 sll \$8,\$8,1	Source	lw \$8, 0x2000(\$0) andi \$9,\$8,1 bne \$9,\$0,odd andi \$8,\$8,0xF	Text Segmen  # load the P into # get the lowest # P is even or or # "6^P mod 17" hav # relative word ar	t  register bit of P dd ? ve 16-number-cycle ddress = P*2	7 0
Bkpt	0x00002000 0x00002020 0x00002040  Address Code 0x00000000 0x8c08200 0x00000004 0x3109000 0x00000000 0x3108000 0x000000010 0x0008404	8 13 10 Basic 0 lw \$8,8192(\$0) 1 andi \$9,\$8,1 5 bne \$9,\$0,5 f andi \$8,\$8,15 0 sll \$8,\$8,1 8 addi \$10,\$0,8200	Source	lw \$8, 0x2000(\$0) andi \$9,\$8,1 bne \$9,\$0,odd andi \$8,\$8,0xF sll \$8,\$8,1	Text Segmen  # load the P into # get the lowest # P is even or or # "6^P mod 17" hav # relative word ar	t  register bit of P dd ? ve 16-number-cycle ddress = P*2 even_result array	7 0
	0x00002000 0x00002020 0x00002040  Address Code 0x00000000 0x8c08200 0x00000000 0x3109000 0x00000000 0x3108000 0x00000010 0x0008404 0x00000014 0x200a200 0x00000018 0x0148502 0x0000001c 0x0800000	8 13 10  Basic 0 lw \$8,8192(\$0) 1 andi \$9,\$8,1 5 bne \$9,\$0,5 f andi \$8,\$8,15 0 sll \$8,\$8,1 8 addi \$10,\$0,8200 0 add \$10,\$10,\$8 d j 0x00000034	Source	lw \$8, 0x2000(\$0) andi \$9,\$8,1 bne \$9,\$0,odd andi \$8,\$8,0xF sll \$8,\$8,1 addi \$10,\$0,0x2008	Text Segmen  # load the P into # get the lowest # P is even or or # "6^P mod 17" hav # relative word ar # the address of ar # the address of ar	t  register bit of P dd ? ve 16-number-cycle ddress = P*2 even_result array answer	7 0
Bkpt	0x00002000 0x00002020 0x00002040  Address Code 0x00000000 0x8c08200 0x00000004 0x3109000 0x00000000 0x1520000 0x00000000 0x3108000 0x00000010 0x0008404 0x00000014 0x200a200 0x00000018 0x0148502	8 13 10  Basic 0 lw \$8,8192(\$0) 1 andi \$9,\$8,1 5 bne \$9,\$0,5 f andi \$8,\$8,15 0 sll \$8,\$8,1 8 addi \$10,\$0,8200 0 add \$10,\$10,\$8 d j 0x00000034	Source	lw \$8, 0x2000(\$0) andi \$9,\$8,1 bne \$9,\$0,odd andi \$8,\$8,0xF sll \$8,\$8,1 addi \$10,\$0,0x2008 add \$10,\$10,\$8	Text Segmen  # load the P into # get the lowest # P is even or or # "6^P mod 17" hav # relative word ar # the address of e	t  register bit of P dd ? ve 16-number-cycle ddress = P*2 even_result array answer	7 0
Bkpt	0x00002000 0x00002020 0x00002040  Address Code 0x00000000 0x8c08200 0x00000000 0x3109000 0x00000000 0x3108000 0x00000010 0x0008404 0x00000014 0x200a200 0x00000018 0x0148502 0x0000001c 0x0800000	8 13 10  Basic 0 lw \$8,8192(\$0) 1 andi \$9,\$8,1 5 bne \$9,\$0,5 f andi \$8,\$8,15 0 sll \$8,\$8,1 8 addi \$10,\$0,8200 0 add \$10,\$10,\$8 d j 0x00000034 f andi \$8,\$8,15	Source	lw \$8, 0x2000(\$0) andi \$9,\$8,1 bne \$9,\$0,odd andi \$8,\$8,0xF sll \$8,\$8,1 addi \$10,\$0,0x2008 add \$10,\$10,\$8 j then andi \$8,\$8,0xF addi \$8,\$8,0xF	Text Segmen  # load the P into # get the lowest # P is even or or # "6^P mod 17" hav # relative word ar # the address of ar # the address of ar	t  register bit of P dd ? ve 16-number-cycle ddress = P*2 even_result array answer ve 16-number-cycle	7 0
Bkpt	0x00002000 0x00002020 0x00002040  Address Code 0x00000000 0x8c08200 0x00000004 0x3109000 0x00000000 0x3108000 0x00000010 0x0008404 0x00000014 0x200a200 0x00000018 0x0148502 0x0000001c 0x0800000 0x0000001c 0x0800000 0x000000000000000000000000000	8 13 10  Basic 0 lw \$8,8192(\$0) 1 andi \$9,\$8,1 5 bne \$9,\$0,5 f andi \$8,\$8,15 0 sll \$8,\$8,1 8 addi \$10,\$0,8200 0 add \$10,\$10,\$8 d j 0x00000034 f andi \$8,\$8,15 f addi \$8,\$8,15	Source 14: 15: 16: 18: 19: 20: 21: 22: 24:	lw \$8, 0x2000(\$0) andi \$9,\$8,1 bne \$9,\$0,odd andi \$8,\$8,0xF sll \$8,\$8,1 addi \$10,\$0,0x2008 add \$10,\$10,\$8 j then andi \$8,\$8,0xF	Text Segmen  # load the P into # get the lowest # P is even or or # "6^P mod 17" hav # relative word ar # the address of ar # "6^P mod 17" hav	t  register bit of P dd ? ve 16-number-cycle ddress = P*2 even_result array answer ve 16-number-cycle	7 0
Bkpt	0x00002000 0x00002040  0x00002040  Address Code 0x00000000 0x8c08200 0x00000004 0x3109000 0x00000000 0x3108000 0x00000010 0x0008404 0x00000014 0x200a200 0x00000015 0x0800000 0x00000016 0x0800000 0x00000016 0x0800000 0x00000020 0x3108000 0x00000020 0x2108fff 0x00000028 0x0008404 0x00000026 0x200a202	8 13 10  Basic 0 lw \$8,8192(\$0) 1 andi \$9,\$8,1 5 bne \$9,\$0,5 f andi \$8,\$8,1 8 addi \$10,\$0,8200 0 add \$10,\$10,\$8 dj 0x0000034 f andi \$8,\$8,15 f addi \$8,\$8,1 8 addi \$10,\$0,8232	Source 14: 15: 16: 18: 19: 20: 21: 22: 24: 25: 26: 27:	lw \$8, 0x2000(\$0) andi \$9,\$8,1 bne \$9,\$0,odd andi \$8,\$8,0xF sll \$8,\$8,1 addi \$10,\$0,0x2008 add \$10,\$10,\$8 j then andi \$8,\$8,0xF addi \$8,\$8,0xF	Text Segmen  # load the P into # get the lowest # P is even or or # "6^P mod 17" hav # relative word ar # the address of a # "6^P mod 17" hav # relative word ar	t  register bit of P dd ? ve 16-number-cycle ddress = P*2 even_result array answer ve 16-number-cycle ddress = (P-1)*2	7 0
Bkpt	0x00002000 0x00002040  0x00002040  Address Code 0x00000000 0x8c08200 0x00000004 0x3109000 0x00000000 0x3108000 0x00000010 0x0008404 0x00000014 0x200a200 0x00000015 0x0800000 0x00000016 0x0800000 0x00000016 0x0800000 0x00000020 0x3108000 0x00000020 0x3108000 0x00000020 0x2108fff 0x00000020 0x2008404 0x00000020 0x2003202 0x00000030 0x0148502	8 13 10  Basic 0 lw \$8,8192(\$0) 1 andi \$9,\$8,1 5 bne \$9,\$0,5 f andi \$8,\$8,1 8 addi \$10,\$0,8200 0 add \$10,\$10,\$8 d j 0x0000034 f andi \$8,\$8,15 f addi \$8,\$8,1 8 addi \$10,\$0,8232 0 add \$10,\$10,\$8	Source 14: 15: 16: 18: 19: 20: 21: 22: 24: 25:	lw \$8, 0x2000(\$0) andi \$9,\$8,1 bne \$9,\$0,odd andi \$8,\$8,0xF sll \$8,\$8,1 addi \$10,\$0,0x2008 add \$10,\$10,\$8 j then andi \$8,\$8,0xF addi \$8,\$8,0xF addi \$8,\$8,1 addi \$10,\$0,0x2028 add \$10,\$10,\$8	Text Segmen  # load the P into # get the lowest # P is even or or # "6^P mod 17" hav # relative word ar # the address of a # "6^P mod 17" hav # relative word ar	t  register bit of P dd ? ve 16-number-cycle ddress = P*2 even_result array answer  ve 16-number-cycle ddress = (P-1)*2  odd_result array	7 0
Bkpt	0x00002000 0x00002040  0x00002040  Address Code 0x00000000 0x8c08200 0x00000000 0x3109000 0x00000000 0x3108000 0x00000010 0x0008404 0x00000010 0x0008404 0x0000010 0x0008404 0x0000010 0x0008404 0x00000010 0x0008404 0x00000010 0x0008404 0x00000010 0x0008404 0x00000010 0x0000000 0x00000010 0x0000000 0x00000010 0x0000000 0x00000010 0x0000000 0x0000000000	Basic 0 lw \$8,8192(\$0) 1 andi \$9,\$8,1 5 bne \$9,\$0,5 f andi \$8,\$8,1 8 addi \$10,\$0,8200 0 add \$10,\$10,\$8 d j 0x0000034 f andi \$8,\$8,1 f addi \$8,\$8,1 8 addi \$10,\$0,8232 0 add \$10,\$10,\$8 0 lw \$11,0(\$10)	Source   14:   15:   16:   18:   19:   20:   21:   22:   24:   25:   26:   27:   28:   30:	lw \$8, 0x2000(\$0) andi \$9,\$8,1 bne \$9,\$0,odd andi \$8,\$8,0xF sll \$8,\$8,1 addi \$10,\$0,0x2008 add \$10,\$10,\$8 j then andi \$8,\$8,0xF addi \$8,\$8,-1 sll \$8,\$8,1 addi \$10,\$0,0x2028 add \$10,\$10,\$8 lw \$11,(\$10)	Text Segmen  # load the P into # get the lowest # P is even or or # "6^P mod 17" hav # relative word ar # the address of a # "6^P mod 17" hav # relative word ar # the address of a	t  register bit of P dd ? ve 16-number-cycle ddress = P*2 even_result array answer  ve 16-number-cycle ddress = (P-1)*2  odd_result array	7 0
Bkpt	0x00002000 0x00002040  0x00002040  Address Code 0x00000000 0x8c08200 0x00000004 0x3109000 0x00000000 0x3108000 0x00000010 0x0008404 0x00000014 0x200a200 0x00000015 0x0800000 0x00000016 0x0800000 0x00000016 0x0800000 0x00000020 0x3108000 0x00000020 0x3108000 0x00000020 0x2108fff 0x00000020 0x2008404 0x00000020 0x2003202 0x00000030 0x0148502	Basic 0 lw \$8,8192(\$0) 1 andi \$9,\$8,1 5 bne \$9,\$0,5 f andi \$8,\$8,1 8 addi \$10,\$0,8200 0 add \$10,\$10,\$8 d j 0x0000034 f andi \$8,\$8,1 f addi \$8,\$8,1 8 addi \$10,\$0,8232 0 add \$10,\$10,\$8 0 lw \$11,0(\$10)	Source 14: 15: 16: 18: 19: 20: 21: 22: 24: 25: 26: 27: 28:	lw \$8, 0x2000(\$0) andi \$9,\$8,1 bne \$9,\$0,odd andi \$8,\$8,0xF sll \$8,\$8,1 addi \$10,\$0,0x2008 add \$10,\$10,\$8 j then andi \$8,\$8,0xF addi \$8,\$8,0xF addi \$8,\$8,1 addi \$10,\$0,0x2028 add \$10,\$10,\$8	Text Segmen  # load the P into # get the lowest # P is even or or # "6^P mod 17" hav # relative word ar # the address of a # "6^P mod 17" hav # relative word ar # the address of a	t  register bit of P dd ? ve 16-number-cycle ddress = P*2 even_result array answer  ve 16-number-cycle ddress = (P-1)*2  odd_result array	7 0
Bkpt	0x00002000 0x00002040  0x00002040  Address Code 0x00000000 0x8c08200 0x00000000 0x3109000 0x00000000 0x3108000 0x00000010 0x0008404 0x00000010 0x0008404 0x0000010 0x0008404 0x0000010 0x0008404 0x00000010 0x0008404 0x00000010 0x0008404 0x00000010 0x0008404 0x00000010 0x0000000 0x00000010 0x0000000 0x00000010 0x0000000 0x00000010 0x0000000 0x0000000000	Basic 0 lw \$8,8192(\$0) 1 andi \$9,\$8,1 5 bne \$9,\$0,5 f andi \$8,\$8,1 8 addi \$10,\$0,8200 0 add \$10,\$10,\$8 d j 0x0000034 f andi \$8,\$8,1 f addi \$8,\$8,1 8 addi \$10,\$0,8232 0 add \$10,\$10,\$8 0 lw \$11,0(\$10)	Source   14:   15:   16:   18:   19:   20:   21:   22:   24:   25:   26:   27:   28:   30:	lw \$8, 0x2000(\$0) andi \$9,\$8,1 bne \$9,\$0,odd andi \$8,\$8,0xF sll \$8,\$8,1 addi \$10,\$0,0x2008 add \$10,\$10,\$8 j then andi \$8,\$8,0xF addi \$8,\$8,-1 sll \$8,\$8,1 addi \$10,\$0,0x2028 add \$10,\$10,\$8 lw \$11,(\$10)	Text Segmen  # load the P into # get the lowest # P is even or or # "6^P mod 17" hav # relative word ar # the address of a # "6^P mod 17" hav # relative word ar # the address of a	t  register bit of P dd ? ve 16-number-cycle ddress = P*2 even_result array answer  ve 16-number-cycle ddress = (P-1)*2  odd_result array	7 0
Bkpt	0x00002000 0x00002040  0x00002040  Address Code 0x00000000 0x8c08200 0x00000000 0x3109000 0x00000000 0x3108000 0x00000010 0x0008404 0x00000010 0x0008404 0x0000010 0x0008404 0x0000010 0x0008404 0x00000010 0x0008404 0x00000010 0x0008404 0x00000010 0x0008404 0x00000010 0x0000000 0x00000010 0x0000000 0x00000010 0x0000000 0x00000010 0x0000000 0x0000000000	Basic 0 lw \$8,8192(\$0) 1 andi \$9,\$8,1 5 bne \$9,\$0,5 f andi \$8,\$8,1 8 addi \$10,\$0,8200 0 add \$10,\$10,\$8 d j 0x0000034 f andi \$8,\$8,1 f addi \$8,\$8,1 8 addi \$10,\$0,8232 0 add \$10,\$10,\$8 0 lw \$11,0(\$10)	Source   14:   15:   16:   18:   19:   20:   21:   22:   24:   25:   26:   27:   28:   30:	lw \$8, 0x2000(\$0) andi \$9,\$8,1 bne \$9,\$0,odd andi \$8,\$8,0xF sll \$8,\$8,1 addi \$10,\$0,0x2008 add \$10,\$10,\$8 j then andi \$8,\$8,0xF addi \$8,\$8,-1 sll \$8,\$8,1 addi \$10,\$0,0x2028 add \$10,\$10,\$8 lw \$11,(\$10)	Text Segmen  # load the P into # get the lowest # P is even or or # "6^P mod 17" hav # relative word ar # the address of a # "6^P mod 17" hav # relative word ar # the address of a	t  register bit of P dd ? ve 16-number-cycle ddress = P*2 even_result array answer  ve 16-number-cycle ddress = (P-1)*2  odd_result array	7 0
Bkpt	0x00002000 0x00002040  0x00002040  Address Code 0x00000000 0x8c08200 0x00000000 0x3109000 0x00000000 0x3108000 0x00000010 0x0008404 0x00000010 0x0008404 0x0000010 0x0008404 0x0000010 0x0008404 0x00000010 0x0008404 0x00000010 0x0008404 0x00000010 0x0008404 0x00000010 0x0000000 0x00000010 0x0000000 0x00000010 0x0000000 0x00000010 0x0000000 0x0000000000	Basic 0 lw \$8,8192(\$0) 1 andi \$9,\$8,1 5 bne \$9,\$0,5 f andi \$8,\$8,1 8 addi \$10,\$0,8200 0 add \$10,\$10,\$8 d j 0x0000034 f andi \$8,\$8,1 f addi \$8,\$8,1 8 addi \$10,\$0,8232 0 add \$10,\$10,\$8 0 lw \$11,0(\$10)	Source   14:   15:   16:   18:   19:   20:   21:   22:   24:   25:   26:   27:   28:   30:	lw \$8, 0x2000(\$0) andi \$9,\$8,1 bne \$9,\$0,odd andi \$8,\$8,0xF sll \$8,\$8,1 addi \$10,\$0,0x2008 add \$10,\$10,\$8 j then andi \$8,\$8,0xF addi \$8,\$8,-1 sll \$8,\$8,1 addi \$10,\$0,0x2028 add \$10,\$10,\$8 lw \$11,(\$10)	Text Segmen  # load the P into # get the lowest # P is even or or # "6^P mod 17" hav # relative word ar # the address of a # "6^P mod 17" hav # relative word ar # the address of a	t  register bit of P dd ? ve 16-number-cycle ddress = P*2 even_result array answer  ve 16-number-cycle ddress = (P-1)*2  odd_result array	7 0
Bkpt	0x00002000 0x00002040  0x00002040  Address Code 0x00000000 0x8c08200 0x00000000 0x3109000 0x00000000 0x3108000 0x00000010 0x0008404 0x00000010 0x0008404 0x0000010 0x0008404 0x0000010 0x0008404 0x00000010 0x0008404 0x00000010 0x0008404 0x00000010 0x0008404 0x00000010 0x0000000 0x00000010 0x0000000 0x00000010 0x0000000 0x00000010 0x0000000 0x0000000000	Basic 0 lw \$8,8192(\$0) 1 andi \$9,\$8,1 5 bne \$9,\$0,5 f andi \$8,\$8,1 8 addi \$10,\$0,8200 0 add \$10,\$10,\$8 d j 0x0000034 f andi \$8,\$8,1 f addi \$8,\$8,1 8 addi \$10,\$0,8232 0 add \$10,\$10,\$8 0 lw \$11,0(\$10)	Source   14:   15:   16:   18:   19:   20:   21:   22:   24:   25:   26:   27:   28:   30:	lw \$8, 0x2000(\$0) andi \$9,\$8,1 bne \$9,\$0,odd andi \$8,\$8,0xF sll \$8,\$8,1 addi \$10,\$0,0x2008 add \$10,\$10,\$8 j then andi \$8,\$8,0xF addi \$8,\$8,-1 sll \$8,\$8,1 addi \$10,\$0,0x2028 add \$10,\$10,\$8 lw \$11,(\$10)	Text Segmen  # load the P into # get the lowest # P is even or or # "6^P mod 17" hav # relative word ar # the address of a # "6^P mod 17" hav # relative word ar # the address of a	t  register bit of P dd ? ve 16-number-cycle ddress = P*2 even_result array answer  ve 16-number-cycle ddress = (P-1)*2  odd_result array	7 0

Value (+8)

9

3

P = 201

P = 1005:

0	<del>-</del>					Text Segment		
Bkpt	Address	Code	Basic	Source				
	0×00000000	0x8c082000	lw \$8,8192(\$0)	14:	lw \$8, 0x2000(\$0)	# load the P into re	egister	
	0×00000004	0×31090001	andi \$9,\$8,1	15:	andi \$9,\$8,1	# get the lowest bi	it of P	
	80000000x0	0×15200005	bne \$9,\$0,5	16:	bne \$9,\$0,odd	# P is even or odd	?	
	0x0000000c	0x3108000f	andi \$8,\$8,15	18:	andi \$8,\$8,0xF	# "6^P mod 17" have	16-number-cycle. So	$P = P \mod 16$
	0×00000010	0×00084040	sll \$8,\$8,1	19:	sll \$8,\$8,1	# relative word addr	ress = P*2	
	0×00000014	0x200a2008	addi \$10,\$0,8200	20:	addi \$10,\$0,0x2008	# the address of eve	en_result array	
	0×00000018	0x01485020	add \$10,\$10,\$8	21:	add \$10,\$10,\$8	# the address of ans	swer	
	0x0000001c	0×0800000d	j 0x00000034	22:	j then			
	0×00000020	0x3108000f	andi \$8,\$8,15	24:	andi \$8,\$8,0xF	# "6^P mod 17" have	16-number-cycle. So	$P = P \mod 16$
	0×00000024	0x2108ffff	addi \$8,\$8,-1	25:	addi \$8,\$8,−1	<pre># relative word addr</pre>	ress = (P-1)*2	
	0x00000028	0×00084040	sll \$8,\$8,1	26:	sll \$8,\$8,1			
	0x0000002c	0x200a2028	addi \$10,\$0,8232	27:	addi \$10,\$0,0x2028	# the address of odd	_result array	
	0×00000030	0x01485020	add \$10,\$10,\$8	28:	add \$10,\$10,\$8	# the address of ans	swer	
	0x00000034	0x8d4b0000	lw \$11,0(\$10)	30:	lw \$11,(\$10)			
	0×00000038	0xac0b2004	sw \$11,8196(\$0)	31:	sw \$11,0x2004(\$0)			
						Data Segment		
Addr	ess	Value (+	0) Value	(+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)
	0×00002	000	1005		10	1	2	4
	0×00002	020	13		9	6	12	7
	0×00002	040	10		3	0	0	0
	0×00002	1060	0		0	0	0	0

• • Ir	nstruction Statistics, V	ersion 1.0	(Ingo Kofler)	
Total:	10			
ALU:	6		60%	
Jump:	0		0%	
Branch:	1		10%	
Memory:	3		30%	
Other:	0		0%	
Tool Control				
Disconnect from MIPS Reset Close				

I found that the value of expression "6^P mod 17" is periodic, and its cycle is 16. The calculation of its value can be divided into two situation depending on P is odd or even. Detailed procedures are attached below.

$6^{P} \mod 17$ .  * P is even $(6^{P}) \mod 17 = (36)^{\frac{P}{2}} \mod 17 = (36 \mod 17)^{\frac{P}{2}} \mod 17$
* P is even  (6) mod 17 = $(3b)^{\frac{P}{2}}$ mod 17 = $(3b)^{\frac{P}{2}}$ mod 17  = $2^{\frac{P}{2}}$ mod 17  index   P   f(P)  0   0   16   1  1   2   18   2  2   4   20   4  3   b   22   8  4   8   24   16  5   10   26   15  6   12     13  7   14     9
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$

Instructions	Registers
lw, andi, bne, sll, addi, add, j, sw	\$8,\$9,\$0,\$10,\$11

### Program II - "Best Match Count"

Code:

.data

```
T: .word 12
2
   best_matching_score: .word -1 # best score = ? within [0, 32]
3
   best_matching_count: .word -1 # how many patterns achieve the best score?
4
   Pattern_Array: .word 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 14, 15, 16, 17, 18, 19, 20
5
6
7
   .text
8
            lw $8, 0x2000($0)
9
            addi $10,$0,0x200C
10
            addi $16,$0,20 # exit traversal flag
11
12
            addi $17,$17,-1
13
                              # the greatest matching score
            addi $15,$15,0
                              # the greatest score counter
14
15
    traversal:
16
           lw $13,0x0($10)
17
           xor $13,$13,$8
18
19
            # count 1's in a 32bit number
20
            xori $13,$13,0xFFFFFFF
21
            srl $14,$13,1
22
23
            andi $13,$13,0x55555555
                                      # imm used as filters to count 1's in a 32-bit binary number
            andi $14,$14,0x55555555
24
            addu $13,$13,$14
25
26
27
            srl $14,$13,2
            andi $13,$13,0x33333333
28
29
            andi $14,$14,0x33333333
            addu $13,$13,$14
30
```

```
srl $14,$13,4
           andi $13,$13,0x07070707
           andi $14,$14,0x07070707
           addu $13,$13,$14
           srl $14,$13,8
           andi $13,$13,0x000F000F
           andi $14,$14,0x000F000F
           addu $13,$13,$14
           srl $14,$13,16
           andi $13,$13,0x0000001F
           andi $14,$14,0x0000001F
                                    # the number of 1's is stored in $13
           addu $13,$13,$14
   count_out:
           beq $17,$13,equal
                               #if both of the two scores are equal
           slt $18,$17,$13
           beq $18,$0,pass
                                  # if current score is greater than biggest score
   greater:
                                   # if current score is greater than biggest score
           add $17,$0,$13
                            # save the biggest score
                              # set the counter to 1
           addi $15,$0,1
           j pass
   equal:
           addi $15,$15,1
58 pass:
           addi $10,$10,4
                            # prepare for next iteration
           addi $16,$16,-1
           beq $16,$0,trave_out
            j traversal
   trave_out:
            sw $17,0x2004($0) # save the greatest matching score
            sw $15,0x2008($0) # save the matching numbers
```

31

32

33

34

35 36

37

38

39

40 41

42

43

44

45 46 47

48

49

50 51

52

53

54

55

56

57

59

60

61

62

63 64 65

66

67 68

### If you wanna run and test it:

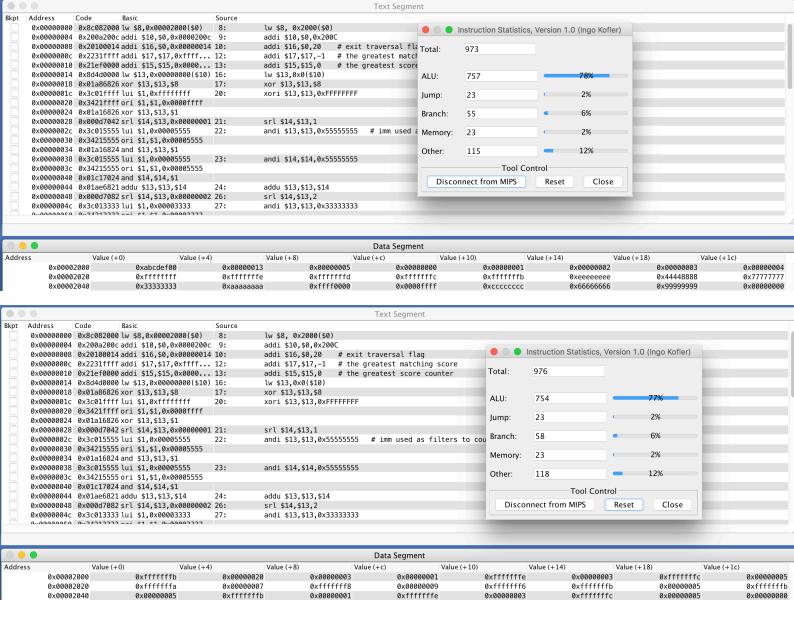
```
.data
T: .word 12
best_matching_score: .word -1 # best score = ? within [0, 32]
best_matching_count: .word -1 # how many patterns achieve the best score?
Pattern_Array: .word 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 14, 15, 16, 17, 18,
19, 20
.text
      lw $8, 0x2000($0)
      addi $10,$0,0x200C
      addi $16,$0,20
                      # exit traversal flag
      addi $17,$17,-1 # the greatest matching score
      addi $15,$15,0
                        # the greatest score counter
traversal:
      lw $13,0x0($10)
      xor $13,$13,$8
      # count 1's in a 32bit number
      xori $13,$13,0xFFFFFFF
      srl $14,$13,1
                               # imm used as filters to count 1's in a 32-bit
      andi $13,$13,0x55555555
binary number
      andi $14,$14,0x55555555
      addu $13,$13,$14
      srl $14,$13,2
      andi $13,$13,0x33333333
      andi $14,$14,0x33333333
      addu $13,$13,$14
      srl $14,$13,4
      andi $13,$13,0x07070707
      andi $14,$14,0x07070707
      addu $13,$13,$14
      srl $14,$13,8
      andi $13,$13,0x000F000F
      andi $14,$14,0x000F000F
      addu $13,$13,$14
```

```
srl $14,$13,16
      andi $13,$13,0x0000001F
      andi $14,$14,0x000001F
      addu $13,$13,$14
                             # the number of 1's is stored in $13
count_out:
      beq $17,$13,equal #if both of the two scores are equal
      slt $18,$17,$13
      beq $18,$0,pass
                            # if current score is greater than biggest score
                         # if current score is greater than biggest score
greater:
      add $17,$0,$13 # save the biggest score
      addi $15,$0,1  # set the counter to 1
      j pass
equal:
      addi $15,$15,1
pass:
      addi $10,$10,4 # prepare for next iteration
      addi $16,$16,-1
      beq $16,$0,trave_out
      j traversal
trave_out:
      sw $17,0x2004(\$0) # save the greatest matching score
      sw $15,0x2008(\$0) # save the matching numbers
```

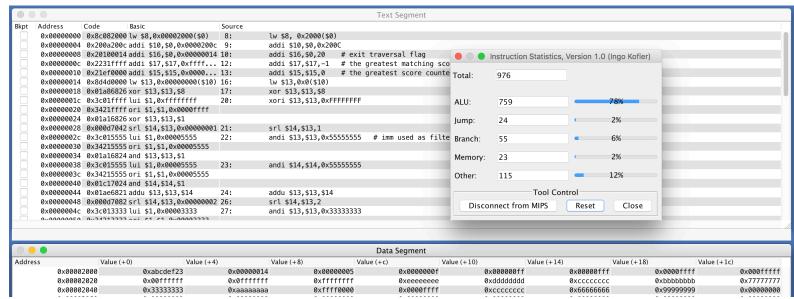
#### Questions:

- 5) level 2 with low DIC
- 6)

#### Tuesday, September 18, 2018



#### T: .word 0xABCDEF23



To find the number of same bits:

- 1. use XOR to find the same bits of two words, for the same bits we got 0, for the different bits we got 1.
- 2. use "result XOR 0xFFFFFFF" to invert the result.
- 3. add the all the bits of that result words together, the value of that words is the number of those same bits.

To record the scores and count the number of best scores:

- 1. Initialize the count with 0, the best score with -1
- 2. For every given word, do the above-mentioned steps, get its score.
- 3. Compare the current score with the recorded best score, if both of them are equal, counter++; if current score is great than recorded best score, set counter to 0, set best score to current score.
- 4. When the traversal is done, store the score and counter as required.

8)

Instructions	Registers
<pre>lw, sw, add, addi, addu, xor, xori, srl, andi, beq, slt, j</pre>	\$0,\$8,\$9,\$10,\$11,\$13,\$14, \$15,\$16,\$17,\$18

9) I can't recall it clearly, maybe two nights. Writing this document indeed cost me a lot of time.

For the program 1, I use something tricky—I do not do a lot of calculation. Instead, I use the periodicity to solve the problem. And I think maybe there are not too much space to optimize.

For the program 2, at first, I use a loop to calculate the sum of every bit of XOR result. The DIC is pretty high, about 4000.

To optimize this problem, I attempt to add counts in a tree pattern, and it make sense. The DIC is under 1000 now.

Talking about how to further optimize it, I don't have a clear idea about that, maybe I can use a more efficient algorithm to calculate the Hamming weight?