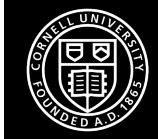
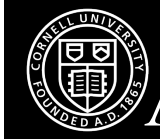


ECE 5725
Embedded Operating Systems
Lecture 11

Prof. Joseph F. Skovira



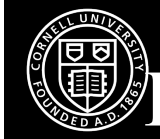
A few items

Lab 2 Week 1

Homework 2

Lab 1 Report

Debug…



Python Basics

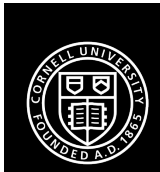
print

input

Convert to str

Convert to int

Concatenate strings



Performance Monitoring with Linux Perf

Two Subsystems

- SYSCALL
- User space tools

Perf installed in kernel

Only need to install perf executable

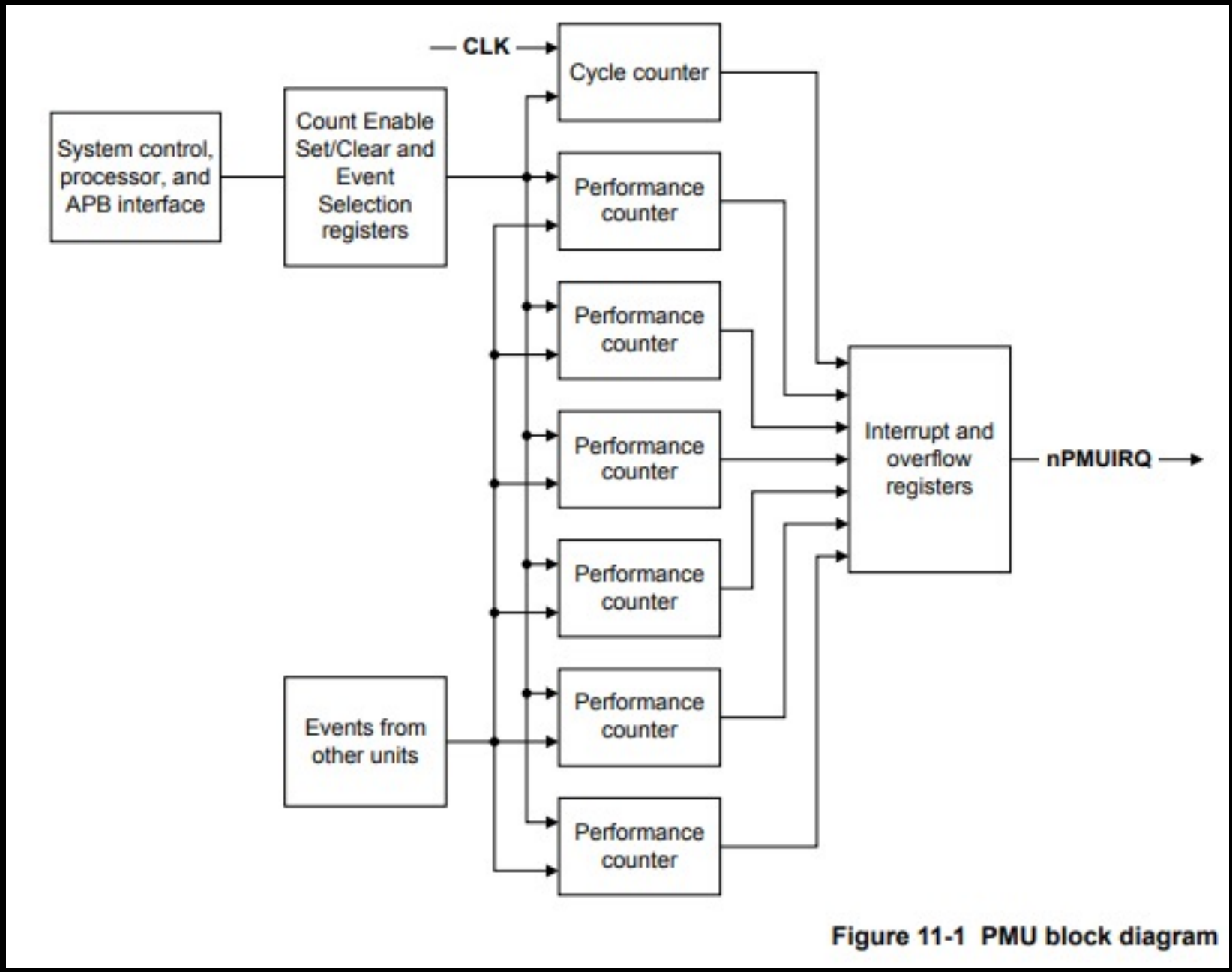
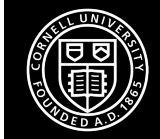


Figure 11-1 PMU block diagram

Table 11-24 PMU events

Event number	Event mnemonic	PMUEVENTx[24:0] bus[cv]	PMU event bus (to trace)[cv]	Event name
0x00	SW_INCR	-	[0]	Instruction architecturally executed (condition check pass) - Software increment
0x01	L1I_CACHE_REFILL	[0]	[1]	Level 1 instruction cache refill
0x02	L1I_TLB_REFILL	[1]	[2]	Level 1 instruction TLB refill
0x03	L1D_CACHE_REFILL	[2]	[3]	Level 1 data cache refill
0x04	L1D_CACHE	-	[5:4]	Level 1 data cache access
0x05	L1D_TLB_REFILL	-	[7:6]	Level 1 data TLB refill
0x08	INST_RETIRED	[6:3]	[11:8]	Instruction architecturally executed
0x09	EXC_TAKEN	[7]	[12]	Exception taken
0x0A	EXC_RETURN	[8]	[13]	Instruction architecturally executed (condition check pass) - Exception return
0x0B	CID_WRITE_RETIRED	-	[14]	Instruction architecturally executed (condition check pass) - Write to CONTEXTIDR
0x10	BR_MIS_PRED	[9]	[15]	Mispredicted or not predicted branch speculatively executed
0x11	CPU_CYCLES	-	[16]	Cycle
0x12	BR_PRED	[10]	[17]	Predictable branch speculatively executed
0x13	MEM_ACCESS	-	[19:18]	Data memory access
0x14	L1I_CACHE	[11]	[20]	Level 1 instruction cache access
0x15	L1D_CACHE_WB	[12]	[21]	Level 1 data cache Write-Back
0x16	L2D_CACHE	-	[23:22]	Level 2 data cache access
0x17	L2D_CACHE_REFILL	[13]	[24]	Level 2 data cache refill
0x18	L2D_CACHE_WB	[14]	[25]	Level 2 data cache Write-Back
0x19	BUS_ACCESS	-	[27:26]	Bus access
0x1A	MEMORY_ERROR	-	[28]	Local memory error
0x1B	INST_SPEC	-	[30:29]	Operation speculatively executed



Performance Monitoring with Linux Perf

stat

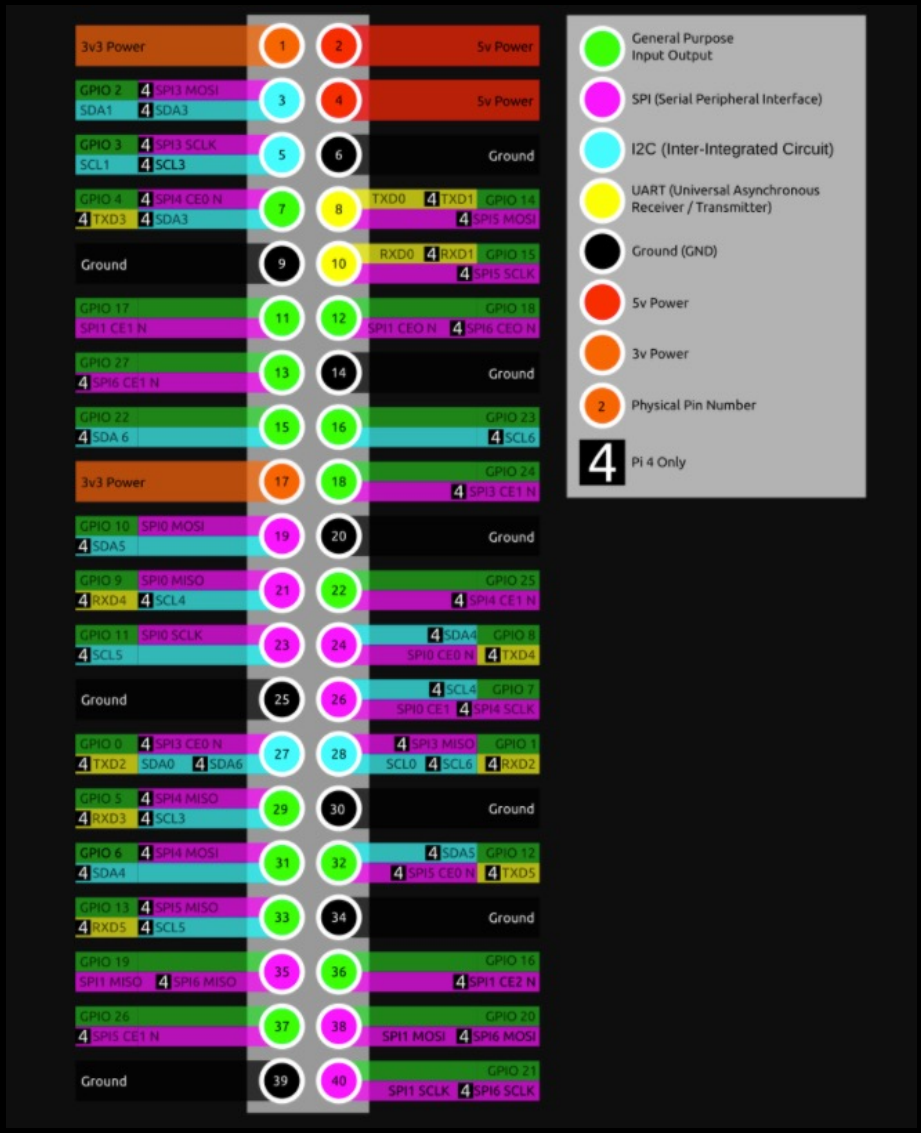
record

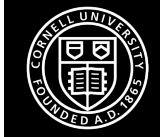
report

top

GPIO

GPIO#	2nd func	pin#		pin#	2nd func	GPIO#
N/A	+3V3	1		2	+5V	N/A
GPIO2	SDA1 (I2C)	3		4	+5V	N/A
GPIO3	SCL1 (I2C)	5		6	GND	N/A
GPIO4	GCLK	7		8	TXD0 (UART)	GPIO14
N/A	GND	9		10	RXD0 (UART)	GPIO15
GPIO17	GEN0	11		12	GEN1	GPIO18
GPIO27	GEN2	13		14	GND	N/A
GPIO22	GEN3	15		16	GEN4	GPIO23
N/A	+3V3	17		18	GEN5	GPIO24
GPIO10	MOSI (SPI)	19		20	GND	N/A
GPIO9	MISO (SPI)	21		22	GEN6	GPIO25
GPIO11	SCLK (SPI)	23		24	CE0_N (SPI)	GPIO8
N/A	GND	25		26	CE1_N (SPI)	GPIO7
(Models A and B stop here)						
EEPROM	ID_SD	27		28	ID_SC	EEPROM
GPIO5	N/A	29		30	GND	N/A
GPIO6	N/A	31		32	-	GPIO12
GPIO13	N/A	33		34	GND	N/A
GPIO19	N/A	35		36	N/A	GPIO16
GPIO26	N/A	37		38	Digital IN	GPIO20
N/A	GND	39		40	Digital OUT	GPIO21



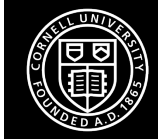


GPIO Events

Polling on GPIO events

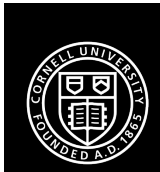
Interrupt on GPIO event

- GPIO Edge Detection
- Trigger python on an event



Python GPIO interrupts

```
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)

GPIO.setup(26, GPIO.IN, pull_up_down=GPIO.PUD_UP)

Try:
    print "Waiting for falling edge on port 26"
    GPIO.wait_for_edge(26, GPIO.FALLING)
    print ”Falling edge detected on port 26”
except KeyboardInterrupt:
    GPIO.cleanup()       # clean up GPIO on CTRL+C exit


GPIO.cleanup()           # clean up GPIO on normal exit
```

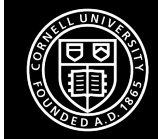


GPIO threaded callback interrupt

Initialize GPIO

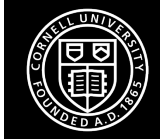
Setup a callback routine

Connect callback to GPIO



Python GPIO interrupts

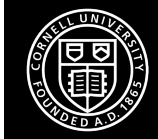
```
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)

GPIO.setup(19, GPIO.IN, pull_up_down=GPIO.PUD_UP)

def GPIO19_callback(channel):
    print "falling edge detected on 19”

# “ main” part of the program
GPIO.add_event_detect(19, GPIO.FALLING, callback=GPIO19_callback)

# Continue on with main processing
#     Background code goes here…

GPIO.cleanup()          # clean up GPIO on normal exit
```



GPIO threaded callback interrupt

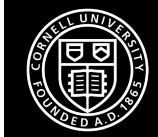
More than one interrupt?

Multiple threaded callbacks



Python GPIO interrupts

```
import RPi.GPIO as GPIO
Import subprocess
GPIO.setmode(GPIO.BCM)

GPIO.setup(19, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(13, GPIO.IN, pull_up_down=GPIO.PUD_UP)

def GPIO19_callback(channel):
    print "falling edge detected on 19”

def GPIO13_callback(channel):
   cmd = 'echo “pause” '
   subprocess.check_output(cmd, shell=True)

# “ main” part of the program
GPIO.add_event_detect(19, GPIO.FALLING, callback=GPIO19_callback)
GPIO.add_event_detect(13, GPIO.FALLING, callback=GPIO13_callback)
# Continue on with main processing – Background code….

GPIO.cleanup()          # clean up GPIO on normal exit
```



Python GPIO interrupts

```
import RPi.GPIO as GPIO
Import subprocess
GPIO.setmode(GPIO.BCM)

GPIO.setup(19, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(13, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(26, GPIO.IN, pull_up_down=GPIO.PUD_UP)
def GPIO19_callback(channel):
    print "falling edge detected on 19”
def GPIO13_callback(channel):
   cmd = 'echo “pause”’ ‘
   subprocess.check_output(cmd, shell=True)
# “ main” part of the program
GPIO.add_event_detect(19, GPIO.FALLING, callback=GPIO19_callback, bouncetime=300)
GPIO.add_event_detect(13, GPIO.FALLING, callback=GPIO13_callback, bouncetime=300)
 Try:
     print "Waiting for falling edge on port 26"
     GPIO.wait_for_edge(26, GPIO.FALLING)
     print ”Falling edge detected on port 26”
 except KeyboardInterrupt:
     GPIO.cleanup()       # clean up GPIO on CTRL+C exit
GPIO.cleanup()           # clean up GPIO on normal exit
```