



# EECS 373

## Design of Microprocessor-Based Systems

Prabal Dutta  
University of Michigan

Lecture 6: Interrupts  
January 27, 2015

Slides developed in part by Mark Brehob

1

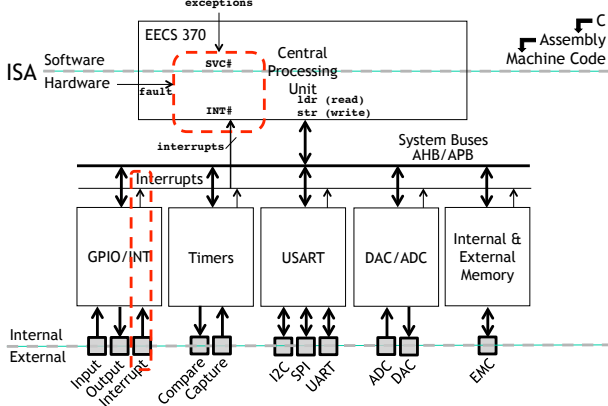
## Announcements



- Additional GSI/IA office hours (OH)
  - Pat Pannuto 10-11am MW in EECS Learning Center
    - (Glass rooms between BBB and Dow)

2

## Interrupts, traps, exceptions, and faults



3

## Interrupts



Merriam-Webster:

- “to break the uniformity or continuity of”

- Informs a program of some external events
- Breaks execution flow

Key questions:

- Where do interrupts come from?
- How do we save state for later continuation?
- How can we ignore interrupts?
- How can we prioritize interrupts?
- How can we share interrupts?

4

## Interrupts



Interrupt (a.k.a. exception or trap):

- An event that causes the CPU to stop executing current program
- Begin executing a special piece of code
  - Called an **interrupt handler** or **interrupt service routine (ISR)**
    - Typically, the ISR does some work
    - Then resumes the interrupted program

Interrupts are really glorified procedure calls, except that they:

- can occur **between** any two instructions
- are “transparent” to the running program (usually)
- are not explicitly requested by the program (typically)
- call a procedure at an address determined by the type of interrupt, not the program

## Two basic types of interrupts (1/2)



- Those caused by an instruction
  - Examples:
    - TLB miss
    - Illegal/unimplemented instruction
    - div by 0
    - SVC (supervisor call, e.g.: SVC #3)
  - Names:
    - Trap, exception

## Two basic types of interrupts (2/2)



- Those caused by the external world
  - External device
  - Reset button
  - Timer expires
  - Power failure
  - System error
- Names:
  - interrupt, external interrupt

## Why are interrupts useful? Example: I/O Data Transfer



Two key questions to determine how data is transferred to/from a non-trivial I/O device:

1. How does the CPU know when data is available?
  - a. Polling
  - b. Interrupts
2. How is data transferred into and out of the device?
  - a. Programmed I/O
  - b. Direct Memory Access (DMA)

## How it works



- Something tells the processor core there is an interrupt
- Core transfers control to code that needs to be executed
- Said code “returns” to old program
- Much harder than it looks.
  - Why?

## Devil is in the details



- How do you figure out *where* to branch to?
- How to you ensure that you can get back to where you started?
- Don't we have a pipeline? What about partially executed instructions?
- What if we get an interrupt while we are processing our interrupt?
- What if we are in a “critical section?”

## Where



- If you know *what* caused the interrupt then you want to jump to the code that handles that interrupt.
  - If you number the possible interrupt cases, and an interrupt comes in, you can just branch to a location, using that number as an offset (this is a branch table)
  - If you don't have the number, you need to *poll* all possible sources of the interrupt to see who caused it.
    - Then you branch to the right code

## Get back to where you once belonged



- Need to store the return address somewhere.
  - Stack *might* be a scary place.
    - *That* would involve a load/store and might cause an interrupt (page fault)!
  - So a dedicated register seems like a good choice
    - But that might cause problems later...
    - What happens if another interrupt happens?

## Modern architectures



- A modern processor has *many* (often 50+) instructions in-flight at once.
  - What do we do with them?
- Drain the pipeline?
  - What if one of them causes an exception?
- Punt all that work
  - Slows us down
- What if the instruction that caused the exception was executed before some other instruction?
  - What if that other instruction caused an interrupt?

## Nested interrupts



- If we get one interrupt while handling another what to do?
  - Just handle it
    - But what about that dedicated register?
    - What if I'm doing something that can't be stopped?
  - Ignore it
    - But what if it is important?
  - Prioritize
    - Take those interrupts you care about. Ignore the rest
    - Still have dedicated register problems.

## Critical section



- We probably need to ignore some interrupts but take others.
  - Probably should be sure *our* code can't cause an exception.
  - Use same prioritization as before.
- What about instructions that shouldn't be interrupted?
  - Disable interrupts while processing an interrupt?

## Our processor



- Over 100 interrupt sources
  - Power on reset, bus errors, I/O pins changing state, data in on a serial bus etc.
- Need a great deal of control
  - Ability to enable and disable interrupt sources
  - Ability to control where to branch to for each interrupt
  - Ability to set interrupt priorities
    - Who wins in case of a tie
    - Can interrupt A interrupt the ISR for interrupt B?
      - If so, A can "preempt" B.
- All that control will involve memory mapped I/O.
  - And given the number of interrupts that's going to be a pain

16

## SmartFusion interrupt sources



Table 7.1 - SmartFusion Interrupt Sources

Cortex-M3 NVIC Input	IRQ Label	IRQ Source
NMI	WDOGTIMEOUT_IRQ	WATCHDOG
INTERRUPT0	WDOGTIMEOUT_IRQ	WATCHDOG
INTERRUPT1	BROWNOUT1_IRQ	VBRPM
INTERRUPT2	BROWNOUT2_IRQ	VBRPM
INTERRUPT3	WDOGTIMEOUT_IRQ	WDOG
INTERRUPT4	PL1_N_IRQ	PL1
INTERRUPT5	EMAC_IRQ	Ethernet MAC
INTERRUPT6	MEM_LBP_IRQ	LBP
INTERRUPT7	ENVM_0_IRQ	ENVM Controller
INTERRUPT8	ENVM_1_IRQ	ENVM Controller
INTERRUPT9	UART_0_IRQ	UART_0
INTERRUPT10	UART_1_IRQ	UART_1
INTERRUPT11	SPI_0_IRQ	SPI_0
INTERRUPT12	SPI_1_IRQ	SPI_1
INTERRUPT13	IC_0_IRQ	IC_0
INTERRUPT14	IC_1_INTERRUPT_IRQ	IC_1
INTERRUPT15	IC_2_INTERRUPT_IRQ	IC_2
INTERRUPT16	IC_3_INTERRUPT_IRQ	IC_3
INTERRUPT17	IC_4_INTERRUPT_IRQ	IC_4
INTERRUPT18	IC_5_INTERRUPT_IRQ	IC_5
INTERRUPT19	IC_6_INTERRUPT_IRQ	IC_6
INTERRUPT20	IC_7_INTERRUPT_IRQ	IC_7
INTERRUPT21	IC_8_INTERRUPT_IRQ	IC_8
INTERRUPT22	IC_9_INTERRUPT_IRQ	IC_9
INTERRUPT23	IC_10_INTERRUPT_IRQ	IC_10
INTERRUPT24	IC_11_INTERRUPT_IRQ	IC_11
INTERRUPT25	IC_12_INTERRUPT_IRQ	IC_12
INTERRUPT26	IC_13_INTERRUPT_IRQ	IC_13
INTERRUPT27	IC_14_INTERRUPT_IRQ	IC_14
INTERRUPT28	IC_15_INTERRUPT_IRQ	IC_15
INTERRUPT29	IC_16_INTERRUPT_IRQ	IC_16
INTERRUPT30	IC_17_INTERRUPT_IRQ	IC_17
INTERRUPT31	IC_18_INTERRUPT_IRQ	IC_18
INTERRUPT32	IC_19_INTERRUPT_IRQ	IC_19
INTERRUPT33	IC_20_INTERRUPT_IRQ	IC_20
INTERRUPT34	IC_21_INTERRUPT_IRQ	IC_21
INTERRUPT35	IC_22_INTERRUPT_IRQ	IC_22
INTERRUPT36	IC_23_INTERRUPT_IRQ	IC_23
INTERRUPT37	IC_24_INTERRUPT_IRQ	IC_24
INTERRUPT38	IC_25_INTERRUPT_IRQ	IC_25
INTERRUPT39	IC_26_INTERRUPT_IRQ	IC_26
INTERRUPT40	IC_27_INTERRUPT_IRQ	IC_27
INTERRUPT41	IC_28_INTERRUPT_IRQ	IC_28
INTERRUPT42	IC_29_INTERRUPT_IRQ	IC_29
INTERRUPT43	IC_30_INTERRUPT_IRQ	IC_30
INTERRUPT44	IC_31_INTERRUPT_IRQ	IC_31
INTERRUPT45	IC_32_INTERRUPT_IRQ	IC_32
INTERRUPT46	IC_33_INTERRUPT_IRQ	IC_33
INTERRUPT47	IC_34_INTERRUPT_IRQ	IC_34
INTERRUPT48	IC_35_INTERRUPT_IRQ	IC_35
INTERRUPT49	IC_36_INTERRUPT_IRQ	IC_36
INTERRUPT50	IC_37_INTERRUPT_IRQ	IC_37
INTERRUPT51	IC_38_INTERRUPT_IRQ	IC_38
INTERRUPT52	IC_39_INTERRUPT_IRQ	IC_39
INTERRUPT53	IC_40_INTERRUPT_IRQ	IC_40
INTERRUPT54	IC_41_INTERRUPT_IRQ	IC_41
INTERRUPT55	IC_42_INTERRUPT_IRQ	IC_42
INTERRUPT56	IC_43_INTERRUPT_IRQ	IC_43
INTERRUPT57	IC_44_INTERRUPT_IRQ	IC_44
INTERRUPT58	IC_45_INTERRUPT_IRQ	IC_45
INTERRUPT59	IC_46_INTERRUPT_IRQ	IC_46
INTERRUPT60	IC_47_INTERRUPT_IRQ	IC_47
INTERRUPT61	IC_48_INTERRUPT_IRQ	IC_48
INTERRUPT62	IC_49_INTERRUPT_IRQ	IC_49
INTERRUPT63	IC_50_INTERRUPT_IRQ	IC_50
INTERRUPT64	IC_51_INTERRUPT_IRQ	IC_51
INTERRUPT65	IC_52_INTERRUPT_IRQ	IC_52
INTERRUPT66	IC_53_INTERRUPT_IRQ	IC_53
INTERRUPT67	IC_54_INTERRUPT_IRQ	IC_54
INTERRUPT68	IC_55_INTERRUPT_IRQ	IC_55
INTERRUPT69	IC_56_INTERRUPT_IRQ	IC_56
INTERRUPT70	IC_57_INTERRUPT_IRQ	IC_57
INTERRUPT71	IC_58_INTERRUPT_IRQ	IC_58
INTERRUPT72	IC_59_INTERRUPT_IRQ	IC_59
INTERRUPT73	IC_60_INTERRUPT_IRQ	IC_60
INTERRUPT74	IC_61_INTERRUPT_IRQ	IC_61
INTERRUPT75	IC_62_INTERRUPT_IRQ	IC_62
INTERRUPT76	IC_63_INTERRUPT_IRQ	IC_63
INTERRUPT77	IC_64_INTERRUPT_IRQ	IC_64
INTERRUPT78	IC_65_INTERRUPT_IRQ	IC_65
INTERRUPT79	IC_66_INTERRUPT_IRQ	IC_66
INTERRUPT80	IC_67_INTERRUPT_IRQ	IC_67
INTERRUPT81	IC_68_INTERRUPT_IRQ	IC_68
INTERRUPT82	IC_69_INTERRUPT_IRQ	IC_69
INTERRUPT83	IC_70_INTERRUPT_IRQ	IC_70
INTERRUPT84	IC_71_INTERRUPT_IRQ	IC_71
INTERRUPT85	IC_72_INTERRUPT_IRQ	IC_72
INTERRUPT86	IC_73_INTERRUPT_IRQ	IC_73
INTERRUPT87	IC_74_INTERRUPT_IRQ	IC_74
INTERRUPT88	IC_75_INTERRUPT_IRQ	IC_75
INTERRUPT89	IC_76_INTERRUPT_IRQ	IC_76
INTERRUPT90	IC_77_INTERRUPT_IRQ	IC_77
INTERRUPT91	IC_78_INTERRUPT_IRQ	IC_78
INTERRUPT92	IC_79_INTERRUPT_IRQ	IC_79
INTERRUPT93	IC_80_INTERRUPT_IRQ	IC_80
INTERRUPT94	IC_81_INTERRUPT_IRQ	IC_81
INTERRUPT95	IC_82_INTERRUPT_IRQ	IC_82
INTERRUPT96	IC_83_INTERRUPT_IRQ	IC_83
INTERRUPT97	IC_84_INTERRUPT_IRQ	IC_84
INTERRUPT98	IC_85_INTERRUPT_IRQ	IC_85
INTERRUPT99	IC_86_INTERRUPT_IRQ	IC_86
INTERRUPT100	IC_87_INTERRUPT_IRQ	IC_87
INTERRUPT101	IC_88_INTERRUPT_IRQ	IC_88
INTERRUPT102	IC_89_INTERRUPT_IRQ	IC_89
INTERRUPT103	IC_90_INTERRUPT_IRQ	IC_90
INTERRUPT104	IC_91_INTERRUPT_IRQ	IC_91
INTERRUPT105	IC_92_INTERRUPT_IRQ	IC_92
INTERRUPT106	IC_93_INTERRUPT_IRQ	IC_93
INTERRUPT107	IC_94_INTERRUPT_IRQ	IC_94
INTERRUPT108	IC_95_INTERRUPT_IRQ	IC_95
INTERRUPT109	IC_96_INTERRUPT_IRQ	IC_96
INTERRUPT110	IC_97_INTERRUPT_IRQ	IC_97
INTERRUPT111	IC_98_INTERRUPT_IRQ	IC_98
INTERRUPT112	IC_99_INTERRUPT_IRQ	IC_99
INTERRUPT113	IC_100_INTERRUPT_IRQ	IC_100
INTERRUPT114	IC_101_INTERRUPT_IRQ	IC_101
INTERRUPT115	IC_102_INTERRUPT_IRQ	IC_102
INTERRUPT116	IC_103_INTERRUPT_IRQ	IC_103
INTERRUPT117	IC_104_INTERRUPT_IRQ	IC_104
INTERRUPT118	IC_105_INTERRUPT_IRQ	IC_105
INTERRUPT119	IC_106_INTERRUPT_IRQ	IC_106
INTERRUPT120	IC_107_INTERRUPT_IRQ	IC_107
INTERRUPT121	IC_108_INTERRUPT_IRQ	IC_108
INTERRUPT122	IC_109_INTERRUPT_IRQ	IC_109
INTERRUPT123	IC_110_INTERRUPT_IRQ	IC_110
INTERRUPT124	IC_111_INTERRUPT_IRQ	IC_111
INTERRUPT125	IC_112_INTERRUPT_IRQ	IC_112
INTERRUPT126	IC_113_INTERRUPT_IRQ	IC_113
INTERRUPT127	IC_114_INTERRUPT_IRQ	IC_114
INTERRUPT128	IC_115_INTERRUPT_IRQ	IC_115
INTERRUPT129	IC_116_INTERRUPT_IRQ	IC_116
INTERRUPT130	IC_117_INTERRUPT_IRQ	IC_117
INTERRUPT131	IC_118_INTERRUPT_IRQ	IC_118
INTERRUPT132	IC_119_INTERRUPT_IRQ	IC_119
INTERRUPT133	IC_120_INTERRUPT_IRQ	IC_120
INTERRUPT134	IC_121_INTERRUPT_IRQ	IC_121
INTERRUPT135	IC_122_INTERRUPT_IRQ	IC_122
INTERRUPT136	IC_123_INTERRUPT_IRQ	IC_123
INTERRUPT137	IC_124_INTERRUPT_IRQ	IC_124
INTERRUPT138	IC_125_INTERRUPT_IRQ	IC_125
INTERRUPT139	IC_126_INTERRUPT_IRQ	IC_126
INTERRUPT140	IC_127_INTERRUPT_IRQ	IC_127
INTERRUPT141	IC_128_INTERRUPT_IRQ	IC_128
INTERRUPT142	IC_129_INTERRUPT_IRQ	IC_129
INTERRUPT143	IC_130_INTERRUPT_IRQ	IC_130
INTERRUPT144	IC_131_INTERRUPT_IRQ	IC_131
INTERRUPT145	IC_132_INTERRUPT_IRQ	IC_132
INTERRUPT146	IC_133_INTERRUPT_IRQ	IC_133
INTERRUPT147	IC_134_INTERRUPT_IRQ	IC_134
INTERRUPT148	IC_135_INTERRUPT_IRQ	IC_135
INTERRUPT149	IC_136_INTERRUPT_IRQ	IC_136
INTERRUPT150	IC_137_INTERRUPT_IRQ	IC_137
INTERRUPT151	IC_138_INTERRUPT_IRQ	IC_138
INTERRUPT152	IC_139_INTERRUPT_IRQ	IC_139
INTERRUPT153	IC_140_INTERRUPT_IRQ	IC_140
INTERRUPT154	IC_141_INTERRUPT_IRQ	IC_141
INTERRUPT155	IC_142_INTERRUPT_IRQ	IC_142
INTERRUPT156	IC_143_INTERRUPT_IRQ	IC_143
INTERRUPT157	IC_144_INTERRUPT_IRQ	IC_144
INTERRUPT158	IC_145_INTERRUPT_IRQ	IC_145
INTERRUPT159	IC_146_INTERRUPT_IRQ	IC_146
INTERRUPT160	IC_147_INTERRUPT_IRQ	IC_147
INTERRUPT161	IC_148_INTERRUPT_IRQ	IC_148
INTERRUPT162	IC_149_INTERRUPT_IRQ	IC_149
INTERRUPT163	IC_150_INTERRUPT_IRQ	IC_150
INTERRUPT164	IC_151_INTERRUPT_IRQ	IC_151
INTERRUPT165	IC_152_INTERRUPT_IRQ	IC_152
INTERRUPT166	IC_153_INTERRUPT_IRQ	IC_153
INTERRUPT167	IC_154_INTERRUPT_IRQ	IC_154
INTERRUPT168	IC_155_INTERRUPT_IRQ	IC_155
INTERRUPT169	IC_156_INTERRUPT_IRQ	IC_156
INTERRUPT170	IC_157_INTERRUPT_IRQ	IC_157
INTERRUPT171	IC_158_INTERRUPT_IRQ	IC_158
INTERRUPT172	IC_159_INTERRUPT_IRQ	IC_159
INTERRUPT173	IC_160_INTERRUPT_IRQ	IC_160
INTERRUPT174	IC_161_INTERRUPT_IRQ	IC_161
INTERRUPT175	IC_162_INTERRUPT_IRQ	IC_162
INTERRUPT176	IC_163_INTERRUPT_IRQ	IC_163
INTERRUPT177	IC_164_INTERRUPT_IRQ	IC_164
INTERRUPT178	IC_165_INTERRUPT_IRQ	IC_165
INTERRUPT179	IC_166_INTERRUPT_IRQ	IC_166
INTERRUPT180	IC_167_INTERRUPT_IRQ	IC_167
INTERRUPT181	IC_168_INTERRUPT_IRQ	IC_168
INTERRUPT182	IC_169_INTERRUPT_IRQ	IC_169
INTERRUPT183	IC_170_INTERRUPT_IRQ	IC_170
INTERRUPT184	IC_171_INTERRUPT_IRQ	IC_171
INTERRUPT185	IC_172_INTERRUPT_IRQ	IC_172
INTERRUPT186	IC_173_INTERRUPT_IRQ	IC_173
INTERRUPT187	IC_174_INTERRUPT_IRQ	IC_174
INTERRUPT188	IC_175_INTERRUPT_IRQ	IC_175
INTERRUPT189	IC_176_INTERRUPT_IRQ	IC_176
INTERRUPT190	IC_177_INTERRUPT_IRQ	IC_177
INTERRUPT191	IC_178_INTERRUPT_IRQ	IC_178
INTERRUPT192	IC_179_INTERRUPT_IRQ	IC_179
INTERRUPT193	IC_180_INTERRUPT_IRQ	IC_180
INTERRUPT194	IC_181_INTERRUPT_IRQ	IC_181
INTERRUPT195	IC_182_INTERRUPT_IRQ	IC_182
INTERRUPT196	IC_183_INTERRUPT_IRQ	IC_183
INTERRUPT197	IC_184_INTERRUPT_IRQ	IC_184
INTERRUPT198	IC_185_INTERRUPT_IRQ	IC_185
INTERRUPT199	IC_186_INTERRUPT_IRQ	IC_186
INTERRUPT200	IC_187_INTERRUPT_IRQ	IC_187
INTERRUPT201	IC_188_INTERRUPT_IRQ	IC_188
INTERRUPT202	IC_189_INTERRUPT_IRQ	IC_189
INTERRUPT203	IC_190_INTERRUPT_IRQ	IC_190
INTERRUPT204	IC_191_INTERRUPT_IRQ	IC_191
INTERRUPT205	IC_192_INTERRUPT_IRQ	IC_192
INTERRUPT206	IC_193_INTERRUPT_IRQ	IC_193
INTERRUPT207	IC_194_INTERRUPT_IRQ	IC_194
INTERRUPT208	IC_195_INTERRUPT_IRQ	IC_195
INTERRUPT209	IC_196_INTERRUPT_IRQ	IC_196
INTERRUPT210	IC_197_INTERRUPT_IRQ	IC_197
INTERRUPT211	IC_198_INTERRUPT_IRQ	IC_198
INTERRUPT212	IC_199_INTERRUPT_IRQ	IC_199
INTERRUPT213	IC_200_INTERRUPT_IRQ	IC_200
INTERRUPT214	IC_201_INTERRUPT_IRQ	IC_201
INTERRUPT215	IC_202_INTERRUPT_IRQ	IC_202
INTERRUPT216	IC_203_INTERRUPT_IRQ	IC_203
INTERRUPT217	IC_204_INTERRUPT_IRQ	IC_204
INTERRUPT218	IC_205_INTERRUPT_IRQ	IC_205
INTERRUPT219	IC_206_INTERRUPT_IRQ	IC_206
INTERRUPT220	IC_207_INTERRUPT_IRQ	IC_207
INTERRUPT221	IC_208_INTERRUPT_IRQ	IC_208
INTERRUPT222	IC_209_INTERRUPT_IRQ	IC_209
INTERRUPT223	IC_210_INTERRUPT_IRQ	IC_210
INTERRUPT224	IC_211_INTERRUPT_IRQ	IC_211
INTERRUPT225	IC_212_INTERRUPT_IRQ	IC_212
INTERRUPT226	IC_213_INTERRUPT_IRQ	IC_213
INTERRUPT227	IC_214_INTERRUPT_IRQ	IC_214
INTERRUPT228	IC_215_INTERRUPT_IRQ	IC_215
INTERRUPT229	IC_216_INTERRUPT_IRQ	IC_216
INTERRUPT230	IC_217_INTERRUPT_IRQ	IC_217
INTERRUPT231	IC_218_INTERRUPT_IRQ	IC_218
INTERRUPT232	IC_219_INTERRUPT_IRQ	IC_219
INTERRUPT233	IC_220_INTERRUPT_IRQ	IC_220
INTERRUPT234	IC_221_INTERRUPT_IRQ	IC_221
INTERRUPT235	IC_222_INTERRUPT_IRQ	IC_222
INTERRUPT236	IC_223_INTERRUPT_IRQ	IC_223
INTERRUPT237	IC_224_INTERRUPT_IRQ	IC_224
INTERRUPT238	IC_225_INTERRUPT_IRQ	IC_225
INTERRUPT239	IC_226_INTERRUPT_IRQ	IC_226
INTERRUPT240	IC_227_INTERRUPT_IRQ	IC_227
INTERRUPT241	IC_228_INTERRUPT_IRQ	IC_228
INTERRUPT242	IC_229_INTERRUPT_IRQ	IC_229
INTERRUPT243	IC_230_INTERRUPT_IRQ	IC_230
INTERRUPT244	IC_231_INTERRUPT_IRQ	IC_231
INTERRUPT245	IC_232_INTERRUPT_IRQ	IC_232
INTERRUPT246	IC_233_INTERRUPT_IRQ	IC_233
INTERRUPT247	IC_234_INTERRUPT_IRQ	IC_234
INTERRUPT248	IC_235_INTERRUPT_IRQ	IC_235
INTERRUPT249	IC_236_INTERRUPT_IRQ	IC_236
INTERRUPT250	IC_237_INTERRUPT_IRQ	IC_237
INTERRUPT251	IC_238_INTERRUPT_IRQ	IC_238
INTERRUPT252	IC_239_INTERRUPT_IRQ	IC_239
INTERRUPT253	IC_240_INTERRUPT_IRQ	IC_240
INTERRUPT254	IC_241_INTERRUPT_IRQ	IC_241
INTERRUPT255	IC_242_INTERRUPT_IRQ	IC_242
INTERRUPT256	IC_243_INTERRUPT_IRQ	IC_243
INTERRUPT257	IC_244_INTERRUPT_IRQ	IC_244
INTERRUPT258	IC_245_INTERRUPT_IRQ	IC_245
INTERRUPT259	IC_246_INTERRUPT_IRQ	IC_246
INTERRUPT260	IC_247_INTERRUPT_IRQ	IC_247
INTERRUPT261	IC_248_INTERRUPT_IRQ	IC_248
INTERRUPT262	IC_249_INTERRUPT_IRQ	IC_249
INTERRUPT263	IC_250_INTERRUPT_IRQ	IC_250
INTERRUPT264	IC_251_INTERRUPT_IRQ	IC_251
INTERRUPT265	IC_252_INTERRUPT_IRQ	IC_252
INTERRUPT266	IC_253_INTERRUPT_IRQ	IC_253
INTERRUPT267	IC_254_INTERRUPT_IRQ	IC_254
INTERRUPT268	IC_255_INTERRUPT_IRQ	IC_255
INTERRUPT269	IC_256_INTERRUPT_IRQ	IC_256
INTERRUPT270	IC_257_INTERRUPT_IRQ	IC_257
INTERRUPT271	IC_258_INTERRUPT_IRQ	IC_258
INTERRUPT272	IC_259_INTERRUPT_IRQ	IC_259
INTERRUPT273	IC_260_INTERRUPT_IRQ	IC_260
INTERRUPT274	IC_261_INTERRUPT_IRQ	IC_261
INTERRUPT275	IC_262_INTERRUPT_IRQ	IC_262
INTERRUPT276	IC_263_INTERRUPT_IRQ	IC_263
INTERRUPT277	IC_264_INTERRUPT_IRQ	IC_264
INTERRUPT278	IC_265_INTERRUPT_IRQ	IC_265
INTERRUPT279	IC_266_INTERRUPT_IRQ	IC_266
INTERRUPT280	IC_267_INTERRUPT_IRQ	IC_267
INTERRUPT281	IC_268_INTERRUPT_IRQ	IC_268
INTERRUPT282	IC_269_INTERRUPT_IRQ	IC_269
INTERRUPT283	IC_270_INTERRUPT_IRQ	IC_270
INTERRUPT284	IC_271_INTERRUPT_IRQ	IC_271
INTERRUPT285	IC_272_INTERRUPT_IRQ	IC_272
INTERRUPT286	IC_273_INTERRUPT_IRQ	IC_273
INTERRUPT287	IC_274_INTERRUPT_IRQ	IC_274
INTERRUPT288	IC_275_INTERRUPT_IRQ	IC_275
INTERRUPT289	IC_276_INTERRUPT_IRQ	IC_276
INTERRUPT290	IC_277_INTERRUPT_IRQ	IC_277
INTERRUPT291	IC_278_INTERRUPT_IRQ	IC_278
INTERRUPT292	IC_279_INTERRUPT_IRQ	IC_279
INTERRUPT293	IC_280_INTERRUPT_IRQ	IC_280
INTERRUPT294	IC_281_INTERRUPT_IRQ	IC_281
INTERRUPT295	IC_282_INTERRUPT_IRQ	IC_282
INTERRUPT296	IC_283_INTERRUPT_IRQ	IC_283
INTERRUPT297	IC_284_INTERRUPT_IRQ	IC_284
INTERRUPT298	IC_285_INTERRUPT_IRQ	IC_285
INTERRUPT299	IC_286_INTERRUPT_IRQ	IC_286
INTERRUPT300	IC_287_INTERRUPT_IRQ	IC_287
INTERRUPT301	IC_288_INTERRUPT_IRQ	IC_288
INTERRUPT302	IC_289_INTERRUPT_IRQ	IC_289
INTERRUPT303	IC_290_INTERRUPT_IRQ	IC_290
INTERRUPT304	IC_291_INTERRUPT_IRQ	IC_291
INTERRUPT305	IC_292_INTERRUPT_IRQ	IC_292
INTERRUPT306	IC_293_INTERRUPT_IRQ	IC_293
INTERRUPT307	IC_294_INTERRUPT_IRQ	IC_294
INTERRUPT308	IC_295_INTERRUPT_IRQ	IC_295
INTERRUPT309	IC_296_INTERRUPT_IRQ	IC_296
INTERRUPT310	IC_297_INTERRUPT_IRQ	IC_297
INTERRUPT311	IC_298_INTERRUPT_IRQ	IC_298
INTERRUPT312	IC_299_INTERRUPT_IRQ	IC_299
INTERRUPT313	IC_300_INTERRUPT_IRQ	IC_300
INTERRUPT314	IC_301_INTERRUPT_IRQ	IC_301
INTERRUPT315	IC_302_INTERRUPT_IRQ	IC_302
INTERRUPT316	IC_303_INTERRUPT_IRQ	IC_303
INTERRUPT317	IC_304_INTERRUPT_IRQ	IC_304
INTERRUPT318	IC_305_INTERRUPT_IRQ	IC_305
INTERRUPT319	IC_306_INTERRUPT_IRQ	IC_306
INTERRUPT320	IC_307_INTERRUPT_IRQ	IC_307
INTERRUPT321	IC_308_INTERRUPT_IRQ	IC_308
INTERRUPT322	IC_309_INTERRUPT_IRQ	IC_309</

## And the interrupt vectors (in startup\_a2fxxxm3.s found in CMSIS, startup\_gcc)



```
g_pfnVectors:
.word _estack
.word Reset_Handler
.word NMI_Handler
.word HardFault_Handler
.word MemManage_Handler
.word BusFault_Handler
.word UsageFault_Handler
.word 0
.word 0
.word 0
.word 0
.word SVC_Handler
.word DebugMon_Handler
.word 0
.word PendSV_Handler
.word SysTick_Handler
.word WdogWakeup_IRQHandler
.word BrownOut_1_5V_IRQHandler
.word BrownOut_3_3V_IRQHandler
..... (they continue)
```

Exception Number	Exception Type	Priority	Description
1	Reset	~3 (Highest)	Reset
2	NMI	~2	Non-maskable interrupt (external NMI input). All fault conditions if the corresponding fault handler is not enabled.
3	Hard fault	~1	Memory management fault: Memory Protection Unit (MPU) violation or access to illegal locations.
4	MemManage fault	Programmable	Bus error: occurs when Advanced High-Performance Bus (AHB) interface receives an error response from a bus slave (also called golden address if it is a data access).
5	Bus fault	Programmable	Exceptions resulting from program error or trying to access coprocessor (the Cortex-M3 does not support a coprocessor).
6	Usage fault	Programmable	Supervisor Call (Debug monitor breakpoints, watchpoints, or external (debug) requests).
7-10	Reserved	NA	Reserved
11	SVC	Programmable	Supervisor Call (Debug monitor breakpoints, watchpoints, or external (debug) requests).
12	Debug monitor	Programmable	Supervisor Call (Debug monitor breakpoints, watchpoints, or external (debug) requests).
13	Reserved	NA	Reserved
14	PendSV	Programmable	Pendable Service Call
15	SysTick	Programmable	System Tick Timer

Exception Number	Exception Type	Priority
16	External interrupt #0	Programmable
17	External interrupt #1	Programmable
...	...	...
255	External interrupt #255	Programmable

19

## How to change where to go on an interrupt? Answer: edit the interrupt vector table [IVT]



```
g_pfnVectors:
.word _estack
.word Reset_Handler
.word NMI_Handler
.word HardFault_Handler
.word MemManage_Handler
.word BusFault_Handler
.word UsageFault_Handler
.word 0
.word 0
.word 0
.word 0
.word SVC_Handler
.word DebugMon_Handler
.word 0
.word PendSV_Handler
.word SysTick_Handler
.word WdogWakeup_IRQHandler
.word BrownOut_1_5V_IRQHandler
.word BrownOut_3_3V_IRQHandler
..... (they continue)
```

```
192 /* =====
193 * Reset_Handler
194 */
195 .global Reset_Handler
196 .type Reset_Handler, %function
197 Reset_Handler:
198 _start:
```

20

## Enabling and disabling interrupt sources



- Interrupt Set Enable and Clear Enable
  - 0xE000E100-0xE000E11C, 0xE000E180-0xE000E19C

0xE000E100	SETENA0	R/W	0	Enable for external interrupt #0-31 bit[0] for interrupt #0 (exception #16) bit[1] for interrupt #1 (exception #17) ... bit[31] for interrupt #31 (exception #47) Write 1 to set bit to 1; write 0 has no effect Read value indicates the current status
0xE000E180	CLRENA0	R/W	0	Clear enable for external interrupt #0-31 bit[0] for interrupt #0 bit[1] for interrupt #1 ... bit[31] for interrupt #31 Write 1 to clear bit to 0; write 0 has no effect Read value indicates the current enable status

21

## Configuring the NVIC (2)



- Set Pending & Clear Pending
  - 0xE000E200-0xE000E21C, 0xE000E280-0xE000E29C

0xE000E200	SETPEND0	R/W	0	Pending for external interrupt #0-31 bit[0] for interrupt #0 (exception #16) bit[1] for interrupt #1 (exception #17) ... bit[31] for interrupt #31 (exception #47) Write 1 to set bit to 1; write 0 has no effect Read value indicates the current status
0xE000E280	CLRPEND0	R/W	0	Clear pending for external interrupt #0-31 bit[0] for interrupt #0 (exception #16) bit[1] for interrupt #1 (exception #17) ... bit[31] for interrupt #31 (exception #47) Write 1 to clear bit to 0; write 0 has no effect Read value indicates the current pending status

22

## Configuring the NVIC (3)



- Interrupt Active Status Register
  - 0xE000E300-0xE000E31C

Address	Name	Type	Reset Value	Description
0xE000E300	ACTIVE0	R	0	Active status for external interrupt #0-31 bit[0] for interrupt #0 bit[1] for interrupt #1 ... bit[31] for interrupt #31
0xE000E304	ACTIVE1	R	0	Active status for external interrupt #32-63
...	...	...	...	...

23

## Interrupt types



- Two main types
  - Level-triggered
  - Edge-triggered

24

## Level-triggered interrupts



- Signaled by asserting a line low or high
- Interrupting device drives line low or high and holds it there until it is serviced
- Device deasserts when directed to or after serviced
- Can share the line among multiple devices (w/ OD+PU)
- Active devices assert the line
- Inactive devices let the line float
- Easy to share line w/o losing interrupts
- But servicing increases CPU load → example
- And requires CPU to keep cycling through to check
- Different ISR costs suggests careful ordering of ISR checks
- Can't detect a new interrupt when one is already asserted

25

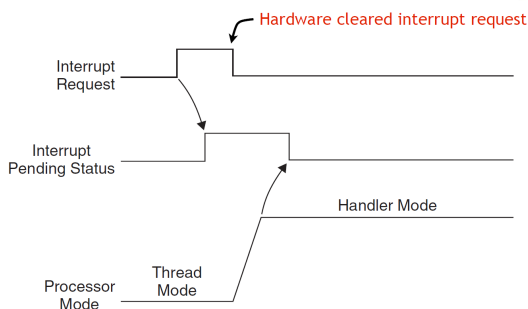
## Edge-triggered interrupts



- Signaled by a level \*transition\* (e.g. rising/falling edge)
- Interrupting device drive a pulse (train) onto INT line
- What if the pulse is too short? Need a pulse extender!
- Sharing \*is\* possible...under some circumstances
- INT line has a pull up and all devices are OC/OD.
- Devices \*pulse\* lines
- Could we miss an interrupt? Maybe...if close in time
- What happens if interrupts merge? Need one more ISR pass
- Must check trailing edge of interrupt
- Easy to detect "new interrupts"
- Benefits: more immune to unserviceable interrupts
- Pitfalls: spurious edges, missed edges
- Source of "lockups" in early computers

26

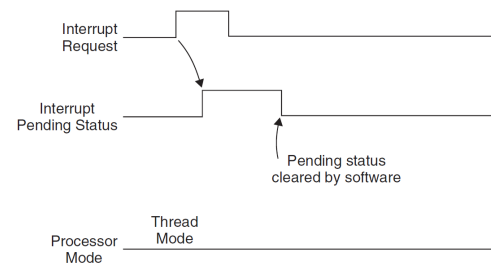
## Pending interrupts



The normal case. Once Interrupt request is seen, processor puts it in "pending" state even if hardware drops the request.  
IPS is cleared by the hardware once we jump to the ISR.

This figure and those following are from *The Definitive Guide to the ARM Cortex-M3*, Section 7.4

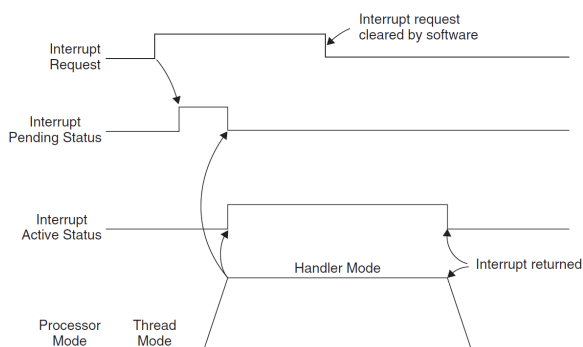
27



In this case, the processor never took the interrupt because we cleared the IPS by hand (via a memory-mapped I/O register)

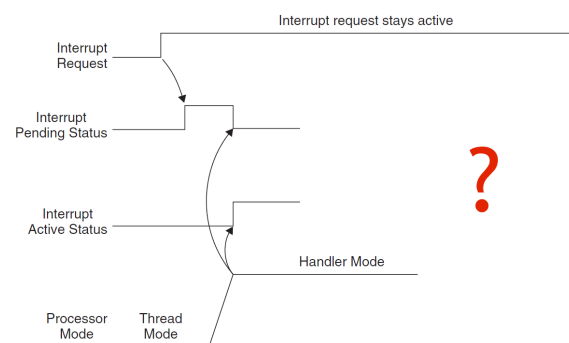
28

## Active Status set during handler execution



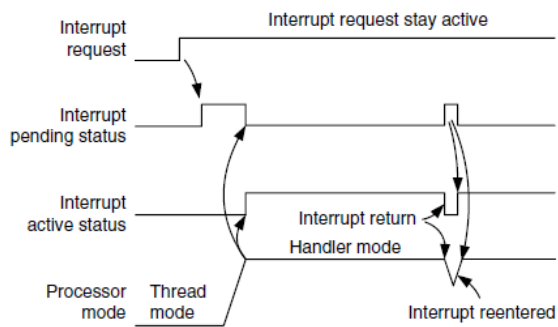
29

## Interrupt Request not Cleared



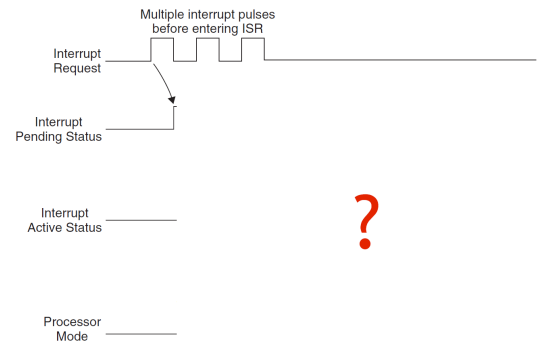
30

## Answer



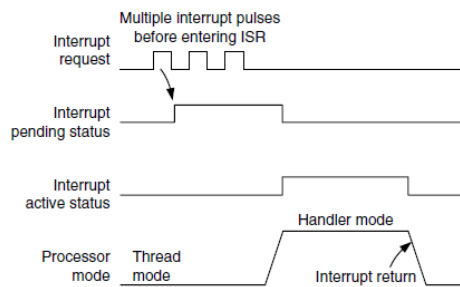
31

## Interrupt pulses before entering ISR



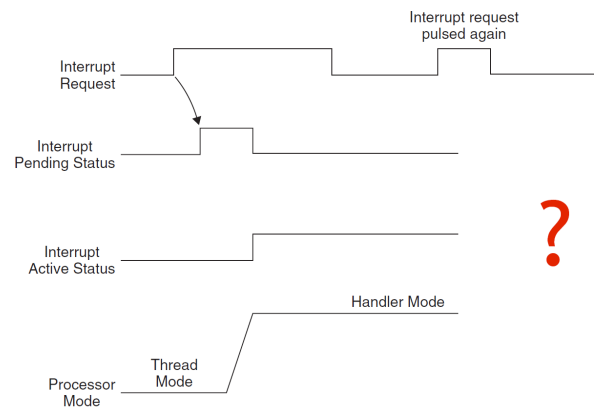
32

## Answer



33

## New Interrupt Request after Pending Cleared



34