INC 361 Microprocessor Systems

by

Pornpoj Hanhaboon

INC361 Ch4 Exceptions and Interrupts

EXCEPTIONS AND INTERRUPTS

THE EVENTS THAT CAN CAUSE CPU TO SUSPEND EXECUTION OF CURRENT PROGRAM AND RESPONSE TO THOSE EVENTS, CPU SHOULD SAVE THE CONTEXT OF THE CURRENT PROGRAM AND CPU STATUS (I.E. CPU REGISTERS) AND TRANSFER TO THE PREDEFINED SERVICE ROUTINES FOR THOSE EVENTS.

THERE ARE TWO CLASSES OF THE EXCEPTIONS :

1 HARDWARE INITIATED

2 SOFTWARE INITIATED.

HARDWARE INITIATED EXCEPTIONS

SOME EVENTS THAT CLASSIFIED AS HARDWARE EXCEPTIONS :

1 CPU RESET

2 HARDWARE FAILURES (E.G. MEMORY OR I/O CIRCUITS FAIL)

3 HARDWARE INTERRUPTS FROM I/O DEVICES.

SOFTWARE INITIATED EXCEPTIONS

SOME EVENTS THAT CLASSIFIED AS SOFTWARE EXCEPTIONS :

1 CPU-DETECTED SOFTWARE ERRORS (E.G. DIVIDED BY ZERO, ADDRESS ERROR)

2 INSTRUCTION-GENERATED EXCEPTIONS (E.G. SOFTWARE-INTERRUPT) .

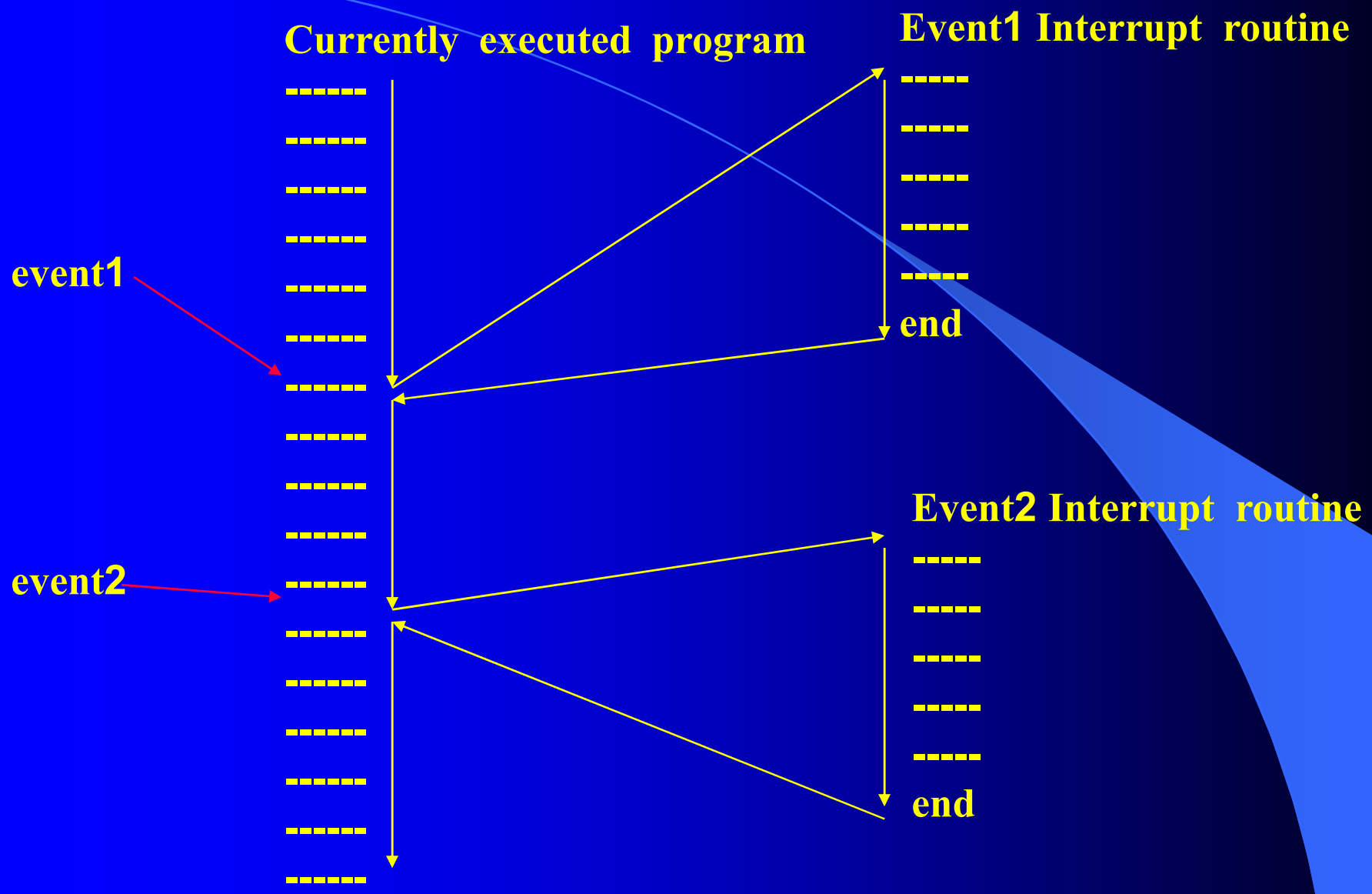
NOTES FOR EXCEPTIONS AND INTERRUPTS

1 SOME MANUFACTURERS DEFINE EXCEPTION EVENTS DIFFERENTLY , SOME CPUs HAVE MANY EXCEPTION EVENTS (ESPECIALLY MODERN CPUs) WHILE SOME CPUs HAVE ONLY FEW EXCEPTION EVENTS

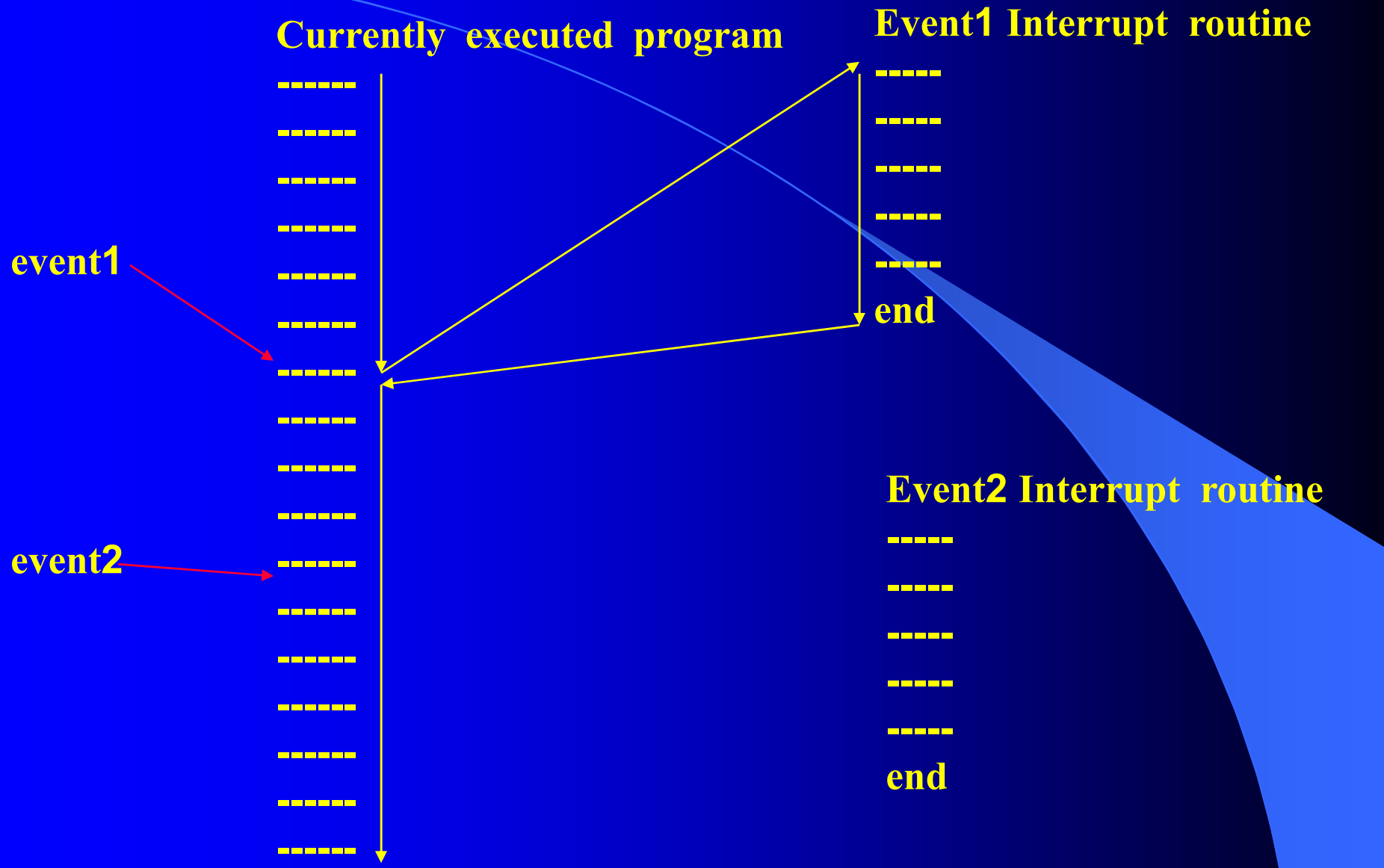
2 SOME CPUs HAVE ONLY HARDWARE INTERRUPT EXCEPTIONS (ESPECIALLY MOST 8-BIT CPUs)

3 THE EXCEPTIONS ARE PROVIDED FOR OPERATING SYSTEM FACILITIES AND SYSTEM CRASH PROTECTION

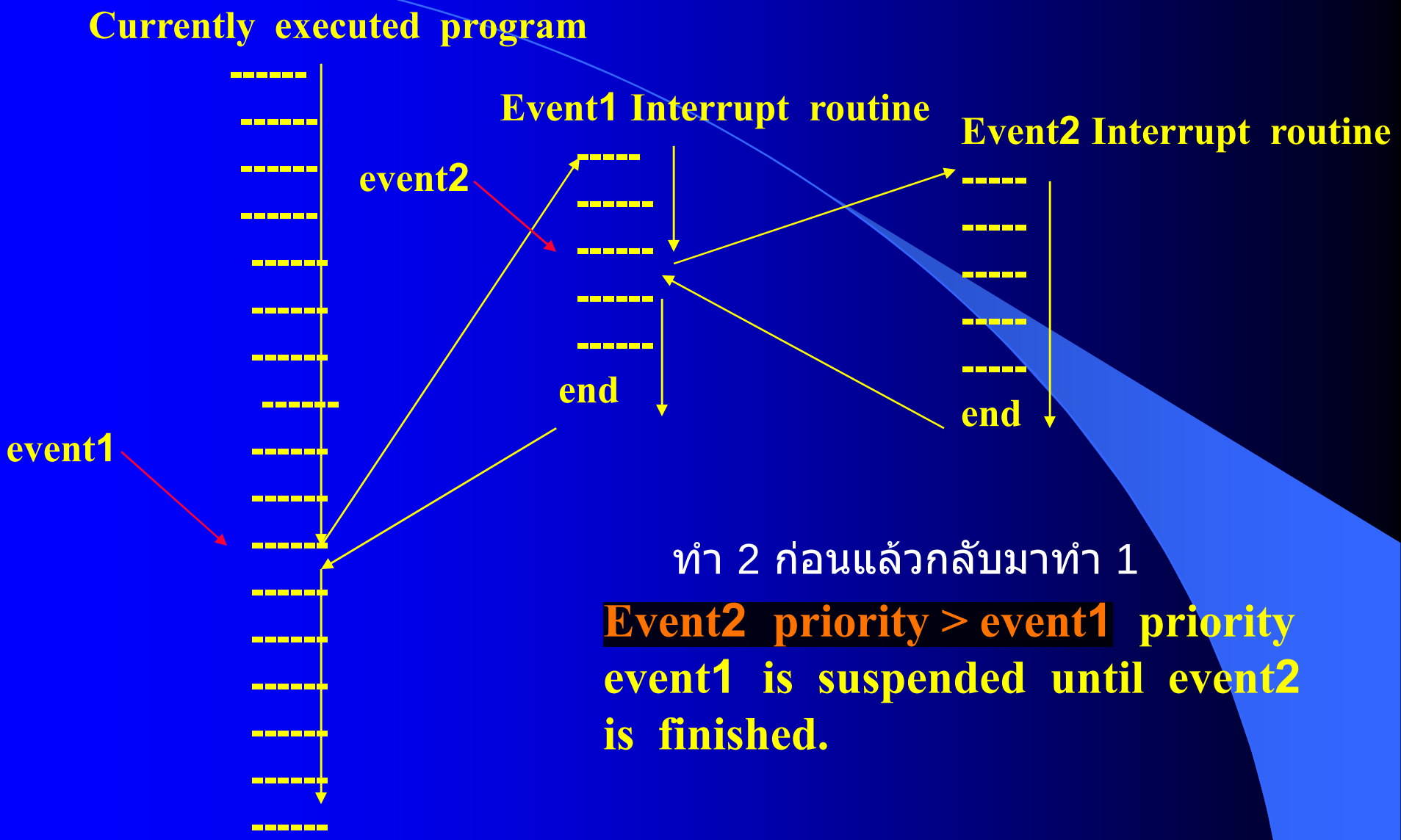
4 YOU CAN FIND THE EXCEPTION EVENTS OF EACH CPU FROM ITS DATA BOOKS.



Both event1 and event2 are enabled.

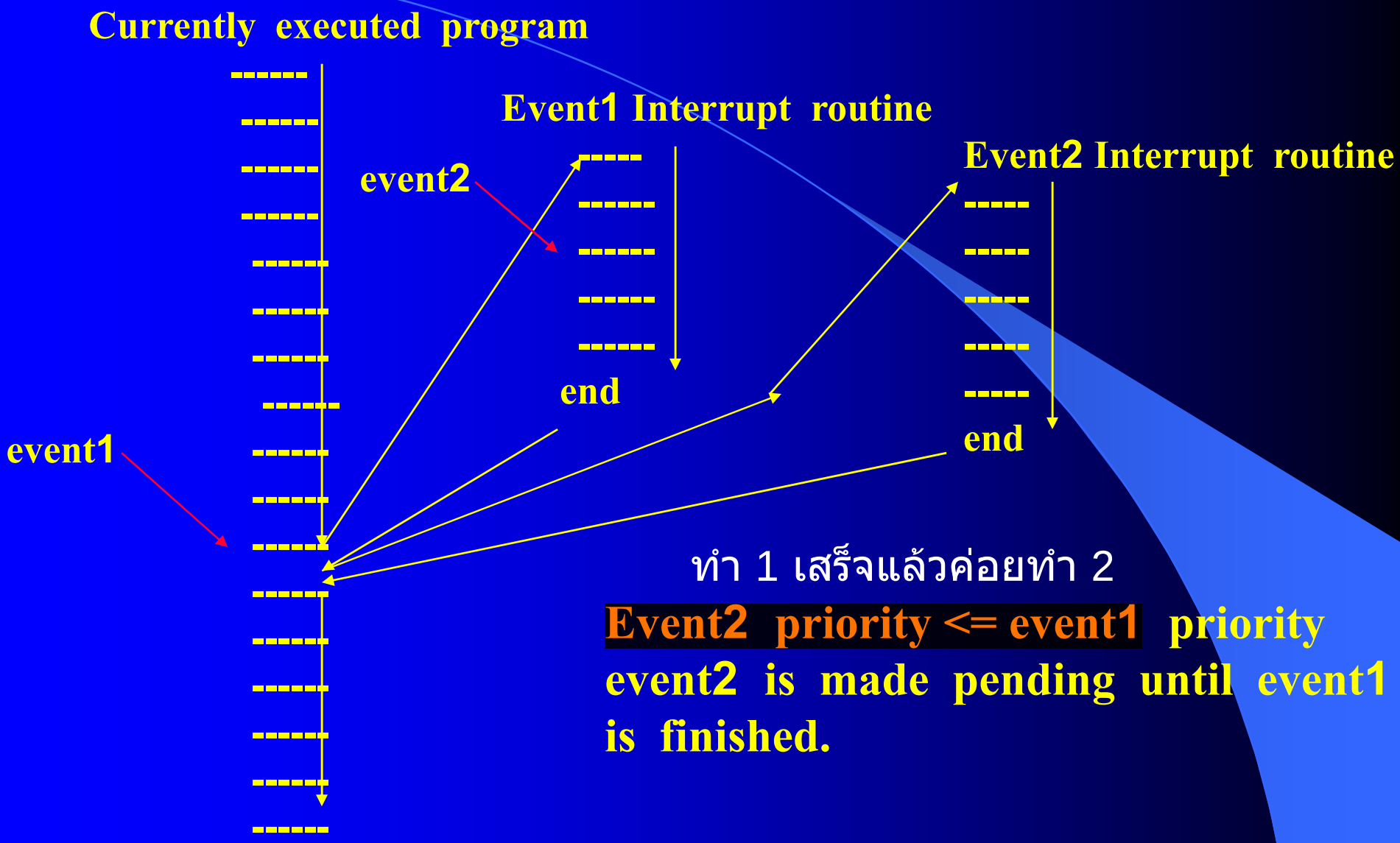


Event1 is enabled and event2 is disabled.



ทำ 2 ก่อนแล้วกลับมาทำ 1

Event2 priority > event1 priority event1 is suspended until event2 is finished.



ทำ 1 เสร็จแล้วค่อยทำ 2

Event2 priority <= event1 priority

event2 is made pending until event1 is finished.

WHAT DO PROGRAMMERS HAVE TO KNOW FOR USING INTERRUPTS ?

1 WHICH **EVENTS** CAN CAUSE INTERRUPTS OF EACH I/O DEVICE?

2 HOW TO **ASSIGNED PRIORITY** OF EACH INTERRUPT REQUEST?

3 HOW TO **ENABLE / DISABLE** INTERRUPTS? global --> each others

สถานะ

4 WHERE IS **INTERRUPT FLAG BIT** OF EACH INTERRUPT ?

5 HOW TO CLEAR EACH INTERRUPT FLAG BIT?

6 WHERE IS ISR OF EACH INTERRUPT REQUEST?

Table 6-2. Exception Vector Assignment

Vectors Numbers		Address		Space[6]	Assignment
Hex	Decimal	Dec	Hex		
0	0	0	000	SP	Reset: Initial SSP[2]
1	1	4	004	SP	Reset: Initial PC[2]
2	2	8	008	SD	Bus Error
3	3	12	00C	SD	Address Error
4	4	16	010	SD	Illegal Instruction
5	5	20	014	SD	Zero Divide
6	6	24	018	SD	CHK Instruction
7	7	28	01C	SD	TRAPV Instruction
8	8	32	020	SD	Privilege Violation
9	9	36	024	SD	Trace
A	10	40	028	SD	Line 1010 Emulator
B	11	44	02C	SD	Line 1111 Emulator
C	12[1]	48	030	SD	(Unassigned, Reserved)
D	13[1]	52	034	SD	(Unassigned, Reserved)
E	14	56	038	SD	Format Error[5]
F	15	60	03C	SD	Uninitialized Interrupt Vector
10–17	16–23[1]	64	040	SD	(Unassigned, Reserved)
		92	05C		—
18	24	96	060	SD	Spurious Interrupt[3]
19	25	100	064	SD	Level 1 Interrupt Autovector
1A	26	104	068	SD	Level 2 Interrupt Autovector
1B	27	108	06C	SD	Level 3 Interrupt Autovector
1C	28	112	070	SD	Level 4 Interrupt Autovector
1D	29	116	074	SD	Level 5 Interrupt Autovector
1E	30	120	078	SD	Level 6 Interrupt Autovector
1F	31	124	07C	SD	Level 7 Interrupt Autovector
20–2F	32–47	128	080	SD	TRAP Instruction Vectors[4]
		188	0BC		—
30–3F	48–63[1]	192	0C0	SD	(Unassigned, Reserved)
		255	0FF		—
40–FF	64–255	256	100	SD	User Interrupt Vectors
		1020	3FC		—

NOTES:

1. Vector numbers 12, 13, 16–23, and 48–63 are reserved for future enhancements by Motorola. No user peripheral devices should be assigned these numbers.
2. Reset vector (0) requires four words, unlike the other vectors which only require two words, and is located in the supervisor program space.
3. The spurious interrupt vector is taken when there is a bus error indication during interrupt processing.
4. TRAP #n uses vector number 32+ n.
5. MC68010 only. This vector is unassigned, reserved on the MC68000 and MC68008.
6. SP denotes supervisor program space, and SD denotes supervisor data space.