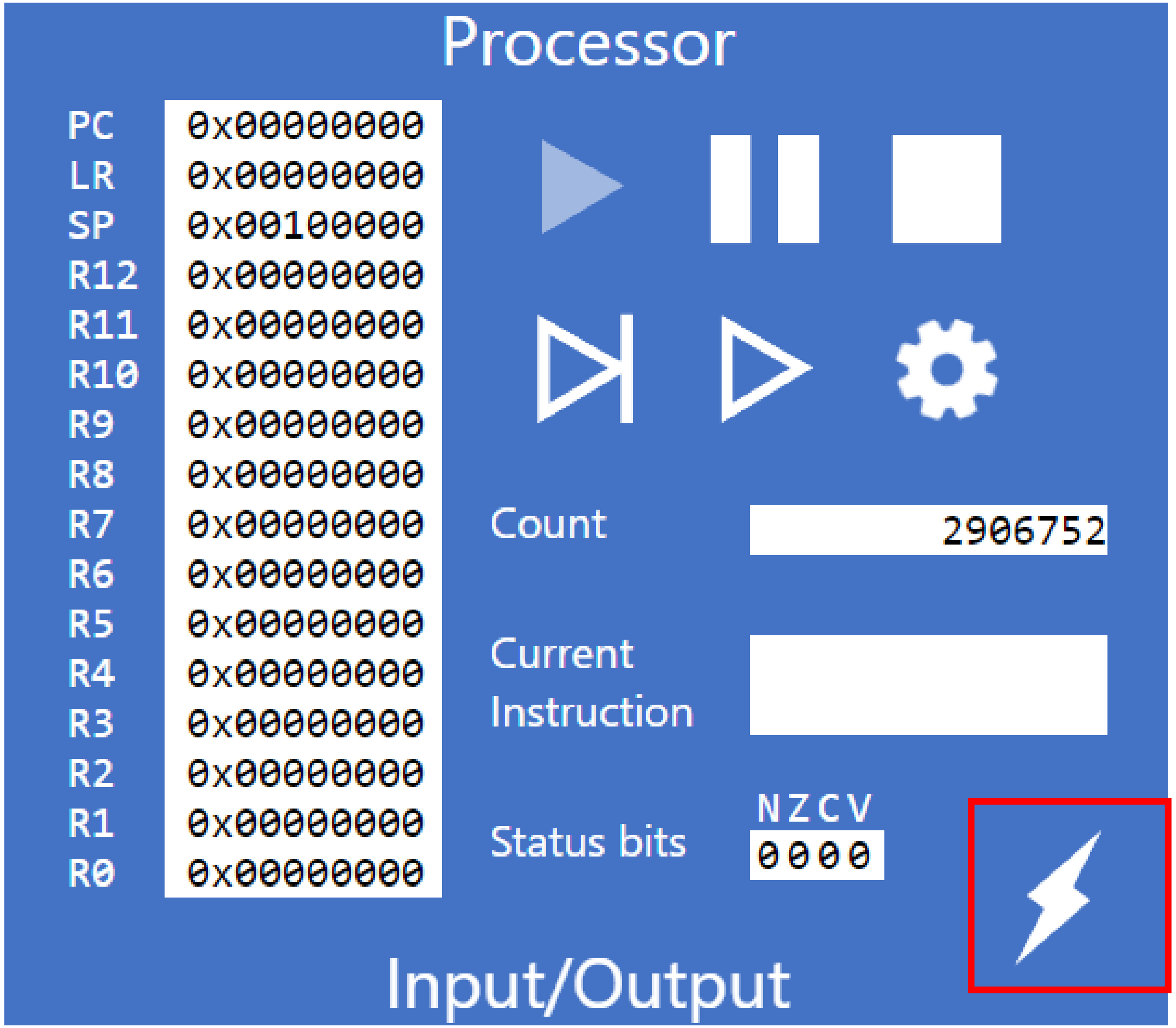
Interrupts in ARMlite

COS10004 Lecture 11.2

Interrupts in ARMlite

- ARMlite offers its own interrupt types
- **Pin Interrupts**:
  - Simulates an Interrupt Request (IRQ) pin
  - Visible on GUI when in “PinISR” mode



Processor

PC	0x00000000
LR	0x00000000
SP	0x00100000
R12	0x00000000
R11	0x00000000
R10	0x00000000
R9	0x00000000
R8	0x00000000
R7	0x00000000
R6	0x00000000
R5	0x00000000
R4	0x00000000
R3	0x00000000
R2	0x00000000
R1	0x00000000
R0	0x00000000

Count 2906752

Current Instruction

Status bits NZCV 0000

Input/Output

Interrupts in ARMlite

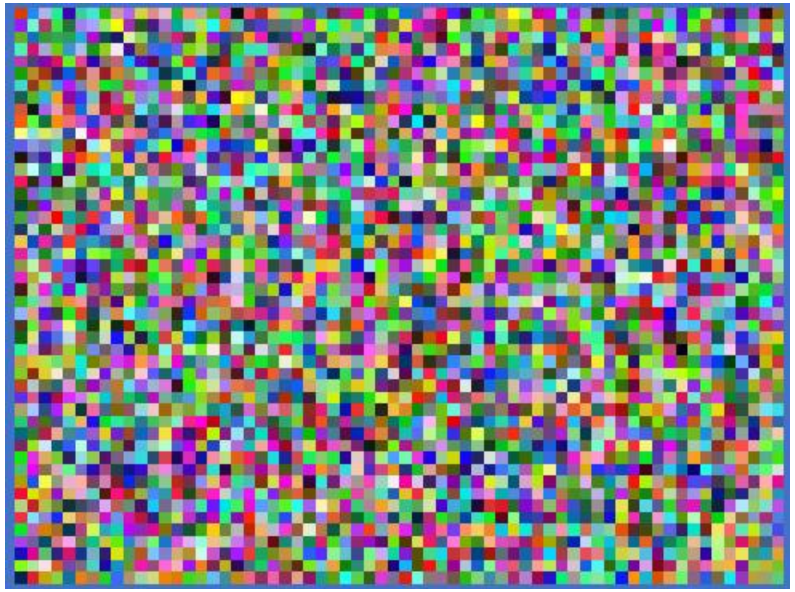
- **Keyboard Interrupts:**
  - High priority interrupts generated when pressing any key
  - This is equivalent to FIQs (Fast Interrupts) in ARM processors.
- **Clock Interrupts:**
  - Many applications need accurate knowledge of time to pace things
  - A hardware timing device can be used to generate interrupts after specified periods, on a regular basis
  - ARMlite simulates this !

Code Examples - Pin Interrupt

- Lets first set up a program that runs continuously generating random pixels in mid-res display mode

```
 //Main program
       MOV R1, #.PixelScreen
       MOV R2, #0        //Pixel index

loop: LDR R0, .Random   //Colour
       STR R0, [R1+R2]
       ADD R2, R2, #4
       CMP R2, #12288
       BLT loop
       MOV R2, #0
       B loop
```



Code Examples - Pin Interrupt

- Lets now modify the program so that everytime the ISR pin is pulled high (ie., clicked), the letter “A” is written to the display.
- To do this, we first need to write an interrupt handler for the event:

```
//Interrupt Routine
writeA:
      PUSH {R0}
      MOV R0, #65
      STR R0, .WriteChar
      POP {R0}
      RFE
```

Code Examples - Pin Interrupt

- Lets now modify the program so that everytime the ISR pin is pulled high (i.e.., clicked), the letter “A” is written to the display.
- To do this, we first need to write an interrupt handler function for the event:

```
//Interrupt Routine
writeA:
      PUSH {R0}
      MOV R0, #65
      STR R0, .WriteChar
      POP {R0}
      RFE
```

RFE (Return From Exception) is a special version of RET for interrupt handling function

Code Examples - Pin Interrupt

- Lastly, we need to add code to the start of our main program to enable Pin ISR interrupt handling:

```
// Set Up Interrupts
 MOV R0, #writeA
 STR R0, .PinISR     //Specify the routine to call when the interrupt Pin is set
 MOV R0, #1
 STR R0, .PinMask                //Enable the interrupt Pin, by setting bit 0 to 1
 STR R0, .InterruptRegister       //Enable interrupts generally
```

Code Examples - Pin Interrupt

- Lastly, we need to add code to the start of our main program to enable Pin ISR interrupt handling:

We write the address of the interrupt handle to the memory mapped register for handling Pin ISR INTs

```
// Set Up Interrupts
 MOV R0, #writeA
 STR R0, .PinISR        //Specify the routine to call when the interrupt Pin is set
 MOV R0, #1
 STR R0, .PinMask                   //Enable the interrupt Pin, by setting bit 0 to 1
 STR R0, .InterruptRegister          //Enable interrupts generally
```

Code Examples - Pin Interrupt

- Lastly, we need to add code to the start of our main program to enable Pin ISR interrupt handling:

```
// Set Up Interrupts
 MOV R0, #writeA
 STR R0, .PinISR     //Specify the routine to call when the interrupt Pin is set
 MOV R0, #1
 STR R0, .PinMask                //Enable the interrupt Pin, by setting bit 0 to 1
 STR R0, .InterruptRegister       //Enable interrupts generally
```

We set the lowest bit to on in both the memory mapped registers .PinMask and .InterruptRegister

Code Examples - Pin Interrupt

- Lets take a look in ARMlite

Code Example – Keyboard interrupt

- Lets now modify our previous program so that instead of handling PinISR interrupts, we handle Keyboard Interrupts.
- While the random pixel display is being continuously upated, we will allow Keyboard inputs to be read and displayed as they occur.
  - That is, we will process them when a keyboard event/interrupt is received.

Code Example – Keyboard interrupt

Lets first write the code for the interrupt handler:

This will simply read the last character entered and display it:

```
//Interrupt routine
writeA:
    PUSH {R0}
    LDR R0, .LastKey
    STR R0, .WriteChar
    POP {R0}
    RFE
```

Code Example – Keyboard interrupt

Lets first write the code for the interrupt handler:

This will simply read the last character entered and display it:

```
//Interrupt routine
writeA:
    PUSH {R0}
    LDR R0, .LastKey
    STR R0, .WriteChar
    POP {R0}
    RFE
```

ARMlite defines a special label referring to address of last entered character

Code Example – Keyboard interrupt

- We also need to enable Keyboard Interrupts at the start of the program:

```
// Set up interrupts
      MOV R0, #writeA
      STR R0, .KeyboardISR
     MOV R0, #1
     STR R0, .KeyboardMask
     STR R0, .InterruptRegister
```

Code Example – Keyboard interrupt

- Full Program:

```
// Set up interrupts
      MOV R0, #writeA
      STR R0, .KeyboardISR
      MOV R0, #1
      STR R0, .KeyboardMask
      STR R0, .InterruptRegister
//Main task - random dots
      MOV R1, #.PixelScreen
      MOV R2, #0        //Pixel index
loop: LDR R0, .Random   //Colour
      STR R0, [R1+R2]
      ADD R2, R2, #4
      CMP R2, #12288
      BLT loop
      MOV R2, #0
      B loop
//Interrupt routine
writeA: PUSH {R0}
      LDR R0, .LastKey
      STR R0, .WriteChar
      POP {R0}
      RFE
```

Code Example – Keyboard interrupt

- Full Program:

Exactly the same main program as last example

```
// Set up interrupts
      MOV R0, #writeA
      STR R0, .KeyboardISR
      MOV R0, #1
      STR R0, .KeyboardMask
      STR R0, .InterruptRegister
//Main task - random dots
      MOV R1, #.PixelScreen
      MOV R2, #0        //Pixel index
loop: LDR R0, .Random   //Colour
      STR R0, [R1+R2]
      ADD R2, R2, #4
      CMP R2, #12288
      BLT loop
      MOV R2, #0
      B loop
//Interrupt routine
writeA: PUSH {R0}
      LDR R0, .LastKey
      STR R0, .WriteChar
      POP {R0}
      RFE
```

Code Example – Keyboard interrupt

- Lets look in ARMLite