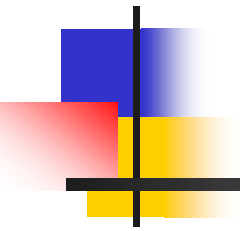
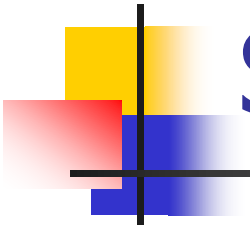


# Interrupts on Arduino Uno

C. H. Ting

August 27, 2011

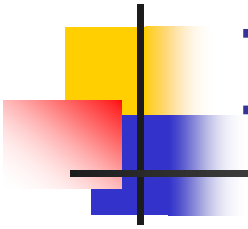
SVFIG



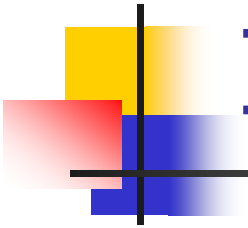
# Summary

- eForth 1 Implementations
- eForth 2 Implementations
- ATmega328P
- AmForth
- 328eForth v2.1
- AVR Studio 4 Tool Suite
- HyperTerminal
- Bill Ragsdale's Applications
- eForth Tutorials
- Conclusion

# Interrupts on ATmega328P



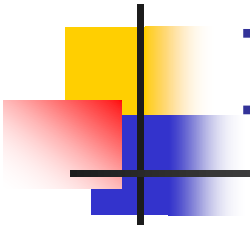
- Interrupt 1 at 0, Reset
- Interrupt 2 at 2, External INT0
- Interrupt 3 at 4, External INT1
- …
- Interrupt 19 at $24, UART RX receive
- …
- Interrupt 26 at $32, SPM Ready



# Interrupts on ATmega328P

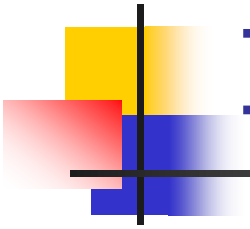
- Global Interrupt Enable, bit 7 in SREG at $5F
- Each interrupting device has an interrupt enable bit in its status register.
- An interrupt cause a CALL to its interrupt service routine, and global interrupt is disabled.
- At the end of the interrupt service routine, a RETI instruction re-enables global interrupt.

# 328eForth Provisions for Interrupts

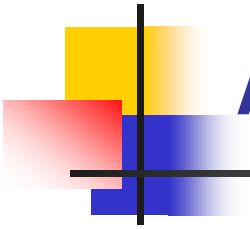


- Flash memory 0-$7F is reserved for reset and interrupts
- Subroutine threading allows interrupt service routines to be written in Forth.
- Only a few more commands are necessary to write interrupt service routines.

# Interrupt.txt

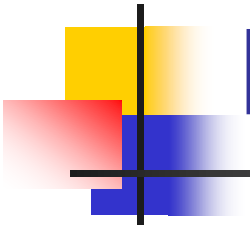


- INIT-VECTORS
- INTERRUPT ( addr # -- )
- RETI,
- SEI,
- CLI,
- NOP,
- PUSH-SREG
- POP-SREG



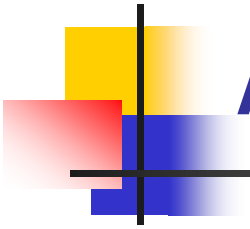
# A Trivial Example

- Interrupt_1.txt
- Use External Interrupt INT0 at PD2 on Pin 4
- On interrupt, increment a variable CNT
- Interrupt on changes at PD2
- PD2 can be changed by writing to PIND2



# Demo

- Download 328eForth v.2.20
- Load interrupt.txt
- Load interrupt_1.txt
- Type:
  - ENABLE
  - ENABLE-INT0
  - CNT ?
  - FLIP CNT ?
  - FLIP CNT ?
  - …



# A Non-Trivial Example

- 328eForth v.2.20 polls UART receiver for KEY.
- 328eForth v2.21 is driven by UART receiver interrupts.
- eForth runs in the background by receiver interrupts.
- Application runs in the foreground task.

# 328eForth220 Interpreter

: QUIT
    Initialize stacks
    Initialize terminal input buffer
    BEGIN QUERY EVAL AGAIN
    ;

QUERY calls ACCEPT

ACCEPT calls KEY

# 328eForth221 Interpreter

```
: QUIT
    Initialize stacks
    Initialize terminal input buffer
    BEGIN NOP AGAIN
    ;
```

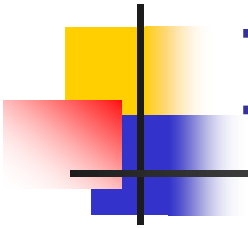
RX interrupts call ACCEPT.

ACCEPT receives keys into terminal input buffer.

At the end of a line, call EVAL.

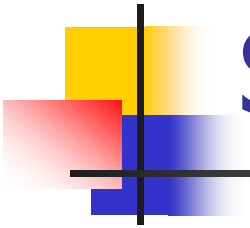
# Interpret.txt, a Test

```
: ACCEPT ( -- )
   KEY ( USART receiver interrupt service )
   DUP 20 - 5f U<
   IF TAP ELSE kTAP THEN
   CUR @ EOT @ =
   IF   CUR @ BOT @ - #TIB !
        0 >IN ! EVAL
        INIT-BUFFER
   THEN
   ;
: TEST INIT-BUFFER BEGIN ACCEPT AGAIN ;
```



# Interrupts Cause Problems

- There are weak points in 328eForth which cannot tolerate interrupts.
- Multiply and divide commands use lots of CPU registers.
- 16 bit data are split into byte registers, and interrupting between byte operations will corrupt data.

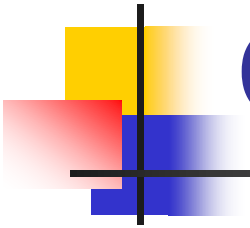


# Solution to Interrupt Problems

- Application running in the foreground must schedule periods allowing interrupts.
- Disable global interrupts when executing normal Forth commands.
- When an error occurred in the background FORTH interpreter, the application will stop. The application can be re-started by reset or ‘0 EXECUTE’.

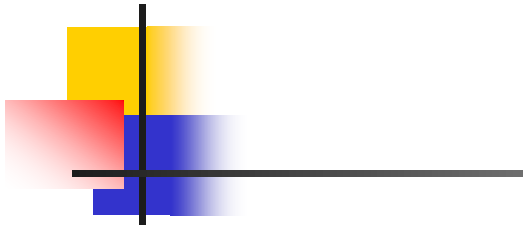
# Demo 2

- Assemble and download 328eForth221.
- Load Interrupt.txt
- Load Traffic_2.txt
- Type:
  - ‘ GO 100 !
  - 100 ERASE
  - 100 100 WRITE
  - 0 EXECUTE ( or push reset button )

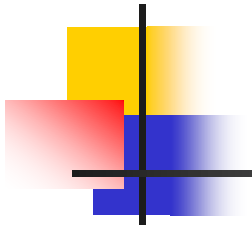


# Conclusion

- Interrupts are not for the faint-hearted.
- 328eForth avoids interrupts, because they confuse ordinary users.
- You can do interrupts in 328eForth, but you have to learn lot more about ATmega328 and Forth before trying.



# Questions?



Thank you.