8-bit 6502 Review

- 8-bit processor
- 3 8-bit registers: A (Accumulator), X, Y (index)
- 16-bit PC
- 8-bit status flag (P): NV-BDIZC
  - Negative, Overflow, Break, Decimal, Interrupt, Zero, Carry
- 8 bit stack pointer (page 1, $0100-$01ff), grows down
• 16-bit data bus (64k) little-endian
• Fast access to Page 0, $0000-$00FF
• Instructions 1-3 bytes
• Chuck Peddle (Maine ’60) co-designer
Addressing Modes

- accumulator: ASL assumes accumulator
- implied: DEX implies X
- immediate: LDA #$12 immediate byte
- absolute: LDA $1234 load from address
- zeropage: LDA $12 loads from zero page
- zeropage,X: LDA $12,X
• zeropage, Y: OPC $12, Y

• absolute, X: LDA $1234, X load from address + X

• absolute, Y: LDA $1234, Y load from address + Y

• indirect: JMP ($1234) jump to address found at $1235/$1234

• indirect, Y-index LDA ($12), Y get address from ZP $13/$12, add Y, load

• X-indexed: indirect: LDA ($12, X) *(ZP+X) (wraps)
• jump relative OPC $BB target is PC + BB
Instruction Set

- ADC – add to A w cary
- AND – and mem with accumulator
- ASL – arithmetic shift left A or mem (by one)
- BCC, BCS – branch on carry clear/set
- BEQ, BNE – branch on equal (Z set)
- BIT – bit test mem w A (result in Z). Also set N and V
to mem bits 7 and 6

- BMI, BPL – branch minus (N set)
- BRK – sw interrupt
- BVC, BVS – branch overflow (V set)
- CLC, CLD, CLI, CLV – clear carry, decimal, interrupt, overflow
- CMP – compare (mem with accumulator)
- CPX, CPY – compare mem with X or Y
• DEC, DEX, DEY – decrement memory, X, Y
• EOR – exclusive or mem (with accumulator)
• INC, INX, INY – increment memory, X, Y
• JMP – jump
• JSR – jump subroutine
• LDA, LDX, LDY – load A, X, Y from mem
• LSR – logical shift right by one
- NOP – no operation
- ORA – or mem with accumulator
- PHA, PLA – push/pop accumulator
- PHP, PLP – push/pop status reg
- ROL, ROR – rotate mem or A left/right
- RTI – return from interrupt
- RTS – return from subroutine
• SBC – subtract memory from A w borrow
• SEC, SED, SEI – set carry, decimal, interrupt
• STA, STX, STY – store A, X, Y to mem
• TAX, TAY – transfer accumulator to X, Y
• TSX, TXS – transfer stack pointer to X
• TXA, TYA – transfer X, Y to accumulator
Apple IIe Enhanced

• Apple II released in 1977

• Apple IIe Platinum released in 1987

• 1MHz 65C02 Processor, 128kB RAM
  (guess about C stands for? How 128kB?)

• 280x192, 6-color graphics (IIe can do DoubleHiRes)

• Press M at boot for menu
• Power: 18 - 20W

• Original machine discrete 7400 logic
Cross-Assembler

- Use ca65 assembler that comes with cc65 compiler
Simulator

• Use linapple simulator
Hello World

putstring code demo
6502 Code Density

- So many 1-byte instructions! Small address! Why isn’t code small?

- 8-bits really too small.

- Awkward emulating 16-bit. 16-bit increment
65c816

- 24-bit addressing (16MB)
- 16-bit stack pointer
- boot in 6502 mode, switch to 16-bit
- X bit: X and Y 16-bit
- M bit: A and Mem 16-bit
- D register – allows setting Zero (Direct) page anywhere
in first 64kb

- Bank register – set which 64kB (of the 16MB) in use at time
65c816 instructions (some on 65c02 as well)

- xce – start 16-bit mode (X bit replaces B) exchange carry and emulation
- xba – exchange top and bottom of A
- mvp/mnv block move instructions (X start, Y dest, A bytes to move) pos/neg
- txy,tyx – move between x and y
- bra – branch always
• inc, dec A

• stz – store zero

• tsb, trb – test and set bit, test and reset bit

• stack relative address mode

• lda ($1234) – load indirect w/o index

• lda $123456 – load with bank

• phx, plx, phy, ply – push/pop X and Y
Discuss the Project