UNDERTALE FOR 64K APPLES: SOURCE FOR SELECTED ASSEMBLY ROUTINES

Due to performance concerns, a large chunk of UnderTale for Apple II relies on 6502 assembly routines. This is particularly important for music synthesis and animation, routines for which we document below.

Accuracy of these code listings is not guaranteed—copy/paste at your own risk. (It's likely safer to refer to the binaries used in the actual demo.)

1. Music synthesis

1.1. spookbit. With music synthesis, it is always important to keep track of timing, which is why you will see cycle counts next to each instruction, as well as the occasional 'screw timing' comment when trying to handle branching.

The routine uses the X and Y registers to keep track of the state of each voice, while the accumulator is alarmingly misused to keep track of which voice needs to be output to the speaker. Note that the Apple's speaker is a 1-bit beeper, and the only way to control it is by accessing `$C030` in memory to toggle the output. Because of this peculiarity, we access the location carefully in each iteration of the loop so that the routine 'knows' the 'state' of the speaker.

```
$0303  A9 FF        LDA #$FF
$0305  85 EB        STA $EB
$0307  AE 01 03     LDX $0301
$030a  AC 02 03     LDY $0302
$030d  86 1E        STX $1E
$030f  84 EE        STY $EE
$0311  A9 00        LDA #$00

$0313  CE 00 03     DEC $0300   [6]
$0316  F0 59        BEQ +$59    [2; 3 if branch to RTS]

$0318  4A           LSR A       [2]
$0319  90 05        BCC +$05    [2 or 3; branches to DEX because screw timing]
$031b  09 80        ORA #$80    [2]
$031d  8D 30 C0     STA $C030   [4]

$0320  CA           DEX         [2]
$0321  D0 04        BNE +$04    [2; 3 if branch to DEY]

$0323  A6 1E        LDX $1E     [3]
$0325  49 AA        EOR #$AA    [2]
```

Date: 2017/09/15.

```
$0327  88              DEY         [2]
$0328  D0 04           BNE +$04    [2; 3 if branch to BCC because screw timing]

$032a  A4 EE           LDY $EE     [3]
$032c  49 55           EOR #$55    [2]

$032e  CA              DEX         [2]
$032f  D0 04           BNE +$04    [2; 3 if branch to DEY]

$0331  A6 1E           LDX $1E     [3]
$0333  49 AA           EOR #$AA    [2]

$0335  88              DEY         [2]
$0336  D0 04           BNE +$04    [2; 3 if branch to BCC because screw timing]

$0338  A4 EE           LDY $EE     [3]
$033a  49 55           EOR #$55    [2]

$033c  90 03           BCC +$03    [2; 3 if branch to LSR]
$033e  8D 30 C0        STA $C030   [4]

$0341  4A              LSR A       [2]
$0342  90 05           BCC +$05    [2 or 3; screw timing]
$0344  09 80           ORA #$80    [2]
$0346  8D 30 C0        STA $C030   [4]

$0349  CA              DEX         [2]
$034a  D0 04           BNE +$04    [2; 3 if branch to DEY]

$034c  A6 1E           LDX $1E     [3]
$034e  49 55           EOR #$55    [2]

$0350  88              DEY         [2]
$0351  D0 04           BNE +$04    [2; 3 if branch to BCC]

$0353  A4 EE           LDY $EE     [3]
$0355  49 AA           EOR #$AA    [2]

$0357  CA              DEX         [2]
$0358  D0 04           BNE +$04    [2; 3 if branch to DEY]

$035a  A6 1E           LDX $1E     [3]
$035c  49 55           EOR #$55    [2]
```

```
$035e  88         DEY        [2]
$035f  D0 04      BNE +$04   [2; 3 if branch to BCC]

$0361  A4 EE      LDY $EE    [3]
$0363  49 AA      EOR #$AA   [2]

$0365  90 03      BCC +$03   [2; 3 if branch to DEC]
$0367  8D 30 C0   STA $C030  [4]

$036a  C6 EB      DEC $EB    [5]
$036c  D0 AA      BNE -$??  [3 if branch back to first LSR A; 2 otherwise]
$036e  4C 13 03   JMP $0313  [3; jump all the way back to DEC $0300]

$0371  60         RTS
```

The result is a two-voice tone generator, which takes pulse widths for each tone at `$0301` and `$0302` and the tone length at `$0300`. The frequencies roughly follow

$$f = \frac{1.023 \times 10^6 \text{ Hz}}{54 + 36 \cdot (\texttt{pulsewidth} - 1)}.$$

1.1.1. *in copy-and-paste-friendly format.*

```
303:A9 FF 85 EB AE 01 03 AC 02 03 86 1E 84 EE A9 00
313:CE 00 03 F0 59
318:4A 90 05 09 80 8D 30 C0
320:CA D0 04 A6 1E 49 AA 88 D0 04 A4 EE 49 55
32e:CA D0 04 A6 1E 49 AA 88 D0 04 A4 EE 49 55 90 03 8D 30 C0
341:4A 90 05 09 80 8D 30 C0
349:CA D0 04 A6 1E 49 55 88 D0 04 A4 EE 49 AA
357:CA D0 04 A6 1E 49 55 88 D0 04 A4 EE 49 AA 90 03 8D 30 C0
36a:C6 EB D0 AA 4C 13 03 60
```

1.1.2. *Applesoft companion code.* Since the routine generates one note/chord at a time and can only take arguments for one note/chord, one way to operate it is by feeding it a numeric data array in Applesoft.

```
 5  DATA   48,164,54,48,164,27,96
    ,130,36,48,138,41,48,138,27,
    96,146,54, 48,164,54,48,164,
    41,48,130,27,48,130,24,48,13
    8,27,48,138,36,96,146,41
 10  FOR I = 1 TO 13: READ A,B,C:
     POKE 768,A: POKE 769,B: POKE
    770,C: CALL 771: NEXT I
```

1.1.3. *octave-switching in real time.* It is possible to slow down the update rate simply by removing some counter updates:

- switch between `32e:ca d0 04` and `4c 35 03` (dex/bne vs jmp)
- switch between `335:88 d0 04` and `4c 3c 03` (dey/bne vs jmp)

- switch between `357:ca d0 04` and `4c 5e 03` (dex/bne vs jmp)
- switch between `35e:88 d0 04` and `4c 65 03` (dey/bne vs jmp)

which extends the pulse width by 1.78, or in alternate terms turns A4 into B3. So to play tunes this slowly, all you need to do is go to this subroutine

```
10030    POKE 814,76: POKE 821,76: POKE 855,76: POKE 862,76
10031    POKE 815,53: POKE 822,60: POKE 856,94: POKE 863,101
10032    POKE 816,3: POKE 823,3: POKE 857,3: POKE 864,3
10033    LPF = 1: RETURN
```

or its assembly equivalent, which is left as an exercise to the reader.

1.1.4. *tempo adjustment in real time.* Sometimes we want finer control of the note duration without altering the pitch. This means that instead of having the duration be multiples of 256 cycles (which is what effectively happens as we decrement EB continuously), we want it to be multiples of, say, 128 or 64.

This actually does not require major adjustment—we just need to specify after the `dec $eb` and `bne` that we want the address to roll over to `#$ff`, for instance. The easiest way to do this would be with `lsr $eb`, which then adds 5 cycles once every 127 loops. This is probably the right order of magnitude to be negligible, and the carry flag set/reset should not persist since we then encounter a `lsr a` instruction.

Here's one way to quadruple the level of control over the tempo:

```
304:3F
317:5F
36e:C6 EB 46 EB 46 EB 4C 13 03 60
```

1.2. **spookbits.** Whereas the spookbit routine accepts parameters for only one possible chord, spookbits is meant to work through many notes all at once. Furthermore, the notes played by one voice do not have to align perfectly with the other, as they would necessarily with spookbit.

1.2.1. *assembly code.* Given that we are now throwing in machinery to handle the voice notes separately, we can no longer hold the entire code in the free part of page 3.

```
$B500: A9 7F        LDA #$7F    [2]
$B502: 85 EB        STA $EB     [3]
$B504: A0 00        LDY #$00    [2]
$B506: 84 E3        STY $E3     [3]
$B508: C8           INY         [2]
$B509: 84 EF        STY $EF     [3]
$B50B: 84 FE        STY $FE     [3]
$B50D: 20 9B B5     JSR $B59B   [6!]
$B510: 20 CB B5     JSR $B5CB   [6!]
$B513: A5 E3        LDA $E3     [3]

$B515: 4A           LSR A       [2]
$B516: 90 05        BCC +$05    [2 or 3; branches to DEX because screw timing]
$B518: 09 80        ORA #$80    [2]
$B51A: 2C 30 C0     BIT $C030   [4]
```

```
$B51D: CA               DEX          [2]
$B51E: D0 04            BNE +$04     [2; 3 if branch to DEY]
$B520: A6 1E            LDX $1E      [3]
$B522: 49 AA            EOR #$AA     [2]

$B524: 88               DEY          [2]
$B525: D0 04            BNE +$04     [2; 3 if branch to BCC because screw timing]
$B527: A4 EE            LDY $EE      [3]
$B529: 49 55            EOR #$55     [2]

$B52B: CA               DEX          [2]
$B52C: D0 04            BNE +$04     [2; 3 if branch to DEY]
$B52E: A6 1E            LDX $1E      [3]
$B530: 49 AA            EOR #$AA     [2]

$B532: 88               DEY          [2]
$B533: D0 04            BNE +$04     [2; 3 if branch to BCC because screw timing]
$B535: A4 EE            LDY $EE      [3]
$B537: 49 55            EOR #$55     [2]

$B539: 90 03            BCC +$03     [2; 3 if branch to LSR]
$B53B: 2C 30 C0         BIT $C030    [4]

$B53E: 4A               LSR A        [2]
$B53F: 90 05            BCC +$05     [2 or 3; screw timing]
$B541: 09 80            ORA #$80     [2]
$B543: 2C 30 C0         BIT $C030    [4]

$B546: CA               DEX          [2]
$B547: D0 04            BNE +$04     [2; 3 if branch to DEY]
$B549: A6 1E            LDX $1E      [3]
$B54B: 49 55            EOR #$55     [2]

$B54D: 88               DEY          [2]
$B54E: D0 04            BNE +$04     [2; 3 if branch to BCC]
$B550: A4 EE            LDY $EE      [3]
$B552: 49 AA            EOR #$AA     [2]

$B554: CA               DEX          [2]
$B555: D0 04            BNE +$04     [2; 3 if branch to DEY]
$B557: A6 1E            LDX $1E      [3]
$B559: 49 55            EOR #$55     [2]

$B55B: 88               DEY          [2]
```

```
$B55C: D0 04          BNE +$04    [2; 3 if branch to BCC]
$B55E: A4 EE          LDY $EE     [3]
$B560: 49 AA          EOR #$AA    [2]

$B562: 90 03          BCC +$03    [2; 3 if branch to DEC]
$B564: 2C 30 C0       BIT $C030   [4]

$B567: C6 EB          DEC $EB     [5]
$B569: D0 AA          BNE -$??    [3 if branch back to first LSR A; 2 otherwise]

$B56B: 85 E3          STA $E3     [3]
$B56D: C6 EC          DEC $EC     [5]
$B56F: D0 0B          BNE +$0B    [2; 3 if branch ahead to DEC $ED]
$B571: A5 EF          LDA $EF     [3]
$B573: F0 07          BEQ +$07    [2; 3 if branch past note load]
$B575: 84 D7          STY $D7     [3]
$B577: 20 9B B5       JSR $B59B   [3; load note 1]
$B57A: A4 D7          LDY $D7     [3]

$B57C: C6 ED          DEC $ED     [5]
$B57E: D0 07          BNE +$07    [2; 3 if branch ahead to LDA $EF]
$B580: A5 FE          LDA $FE     [3]
$B582: F0 03          BEQ +$03    [2; 3 if branch past note load]
$B584: 20 CB B5       JSR $B5CB   [3; load note 2]

$B587: A5 EF          LDA $EF     [3]
$B589: 05 FE          ORA $FE     [3]
$B58B: D0 01          BNE +$01    [2; 3 if branch past RTS]
$B58D: 60             RTS         [6]

$B58E: A9 7F          LDA #$7F    [2]
$B590: 85 EB          STA $EB     [3]
$B592: A5 E3          LDA $E3     [3]
$B594: 4C 15 B5       JMP $B515   [3; jump back to first LSR A]

$B59B: A0 00          LDY #$00    [2]
$B59D: B1 FA          LDA ($FA),Y [5-6]
$B59F: D0 0C          BNE +$0C    [2; 3 if branch to other STA $1E]
$B5A1: AA             TAX         [2]
$B5A2: CA             DEX         [2]
$B5A3: 86 1E          STX $1E     [3]
$B5A5: 8D 1C B5       STA $B51C   [4; overwrite first C0]
$B5A8: 8D 3D B5       STA $B53D   [4; overwrite second C0]
$B5AB: D0 0B          BNE +$0B    [3; must branch to INY]
$B5AD: 85 1E          STA $1E     [3]
```

```
$B5AF: AA          TAX         [2]
$B5B0: A9 C0       LDA #$C0    [2]
$B5B2: 8D 1C B5    STA $B51C   [4; rewrite first C0]
$B5B5: 8D 3D B5    STA $B53D   [4; rewrite second C0]
$B5B8: C8          INY         [2]
$B5B9: B1 FA       LDA ($FA),Y [5-6]
$B5BB: 85 EC       STA $EC     [3]
$B5BD: 85 EF       STA $EF     [3]
$B5BF: A5 FA       LDA $FA     [3]
$B5C1: 18          CLC         [2]
$B5C2: 69 02       ADC #$02    [2]
$B5C4: 90 02       BCC +$02    [2; 3 if branch past INC]
$B5C6: E6 FB       INC $FB     [5]
$B5C8: 85 FA       STA $FA     [3]
$B5CA: 60          RTS         [6]

$B5CB: A0 00       LDY #$00    [2]
$B5CD: B1 FC       LDA ($FC),Y [5-6]
$B5CF: D0 0C       BNE +$0C    [2; 3 if branch to other STA $1E]
$B5D1: 85 EE       STA $EE     [3]
$B5D3: C6 EE       DEC $EE     [5]
$B5D5: 8D 45 B5    STA $B545   [4; overwrite third C0]
$B5D8: 8D 66 B5    STA $B566   [4; overwrite fourth C0]
$B5DB: D0 0B       BNE +$0B    [3; must branch to INY]
$B5DD: 85 EE       STA $EE     [3]
$B5DF: EA          NOP         [2]
$B5E0: A9 C0       LDA #$C0    [2]
$B5E2: 8D 45 B5    STA $B545   [4; rewrite third C0]
$B5E5: 8D 66 B5    STA $B566   [4; rewrite fourth C0]
$B5E8: C8          INY         [2]
$B5E9: B1 FC       LDA ($FC),Y [5-6]
$B5EB: 85 ED       STA $ED     [3]
$B5ED: 85 FE       STA $FE     [3]
$B5EF: A5 FC       LDA $FC     [3]
$B5F1: 18          CLC         [2]
$B5F2: 69 02       ADC #$02    [2]
$B5F4: 90 02       BCC +$02    [2; 3 if branch past INC]
$B5F6: E6 FD       INC $FD     [5]
$B5F8: 85 FC       STA $FC     [3]
$B5FA: A4 EE       LDY $EE     [3]
$B5FC: 60          RTS         [6]
```

1.2.2. *test example.*

```
b600: c3 0c 00 10 c3 18 00 04 c3 0e c3 18 00 04 c3 18
b610: 00 04 c3 18 00 04 c3 0e c3 0e c3 0e c3 0e 00 0e
b620: db 0c 00 10 db 18 00 04 db 0e db 18 00 04 db 18
```

```
b630: 00 04 db 18 00 04 db 0e db 0e db 0e db 0e 00 0e
b640: e8 0c 00 10 e8 18 00 04 e8 0e e8 18 00 04 e8 18
b650: 00 04 e8 18 00 04 e8 0e e8 0e e8 0e e8 0e 00 0e
b660: f6 0c 00 10 f6 18 00 04 f6 0e f6 18 00 04 db 18
b670: 00 04 db 18 00 04 db 0e db 0e db 0e db 0e 00 0e
b680: 00 00
b690: 61 0c 00 02 61 0c 00 02 30 18 00 04 41 18 00 12
b6a0: 45 18 00 04 49 18 00 04 52 18 00 04 61 0e 52 0e 49 0e
b6b2: 6d 0c 00 02 6d 0c 00 02 30 18 00 04 41 18 00 12
b6c2: 45 18 00 04 49 18 00 04 52 18 00 04 61 0e 52 0e 49 0e
b6d4: 74 0c 00 02 74 0c 00 02 30 18 00 04 41 18 00 12
b6e4: 45 18 00 04 49 18 00 04 52 18 00 04 61 0e 52 0e 49 0e
b6f6: 7b 0c 00 02 7b 0c 00 02 30 18 00 04 41 18 00 12
b706: 45 18 00 04 49 18 00 04 52 18 00 04 61 0e 52 0e 49 0e
b718: 00 00
00fa: 00 b6 90 b6
b500g
```

It may be apparent that the only arguments that this routine takes are the start addresses for the note data to feed into each voice (at `$fa` and `$fc`). Each note has a byte indicating duration and a byte indicating pulse width, and if both are zero the routine stops reading in any further notes. Furthermore, this engine actually accommodates rests, which spookbit does not.

Nonetheless, spookbit is in the final diskette alongside spookbits (which was originally intended to supersede spookbit altogether—hence the confusing name) because of certain applications where it is actually quite useful to have the ability to generate only one arbitrary note instead of a sequence that has to be played all at once.

1.2.3. *copy-paste friendly hex code.* This also removes the extraneous NOP at `$B5DF`.

```
B500: A9 7F 85 EB A0 00 84 E3 C8 84 EF 84 FE
B50D: 20 9B B5 20 CB B5 A5 E3
B515: 4A 90 05 09 80 2C 30 C0
B51D: CA D0 04 A6 1E 49 AA 88 D0 04 A4 EE 49 55
B52B: CA D0 04 A6 1E 49 AA 88 D0 04 A4 EE 49 55
B539: 90 03 2C 30 C0
B53E: 4A 90 05 09 80 2C 30 C0
B546: CA D0 04 A6 1E 49 55 88 D0 04 A4 EE 49 AA
B554: CA D0 04 A6 1E 49 55 88 D0 04 A4 EE 49 AA
B562: 90 03 2C 30 C0
B567: C6 EB D0 AA 85 E3
B56D: C6 EC D0 0B A5 EF F0 07 84 D7 20 9B B5 A4 D7
B57C: C6 ED D0 07 A5 FE F0 03 20 CB B5
B587: A5 EF 05 FE D0 01
B58D: 60
B58E: A9 7F 85 EB A5 E3 4C 15 B5
```

```
B59B: A0 00 B1 FA D0 0C
B5A1: AA CA 86 1E 8D 1C B5 8D 3D B5 D0 0B
B5AD: 85 1E AA A9 C0 8D 1C B5 8D 3D B5
B5B8: C8 B1 FA 85 EC 85 EF A5 FA
B5C1: 18 69 02 90 02 E6 FB 85 FA
B5CA: 60

B5CB: A0 00 B1 FC D0 0C
B5D1: 85 EE C6 EE 8D 45 B5 8D 66 B5 D0 0A
B5DD: 85 EE A9 C0 8D 45 B5 8D 66 B5
B5E7: C8 B1 FC 85 ED 85 FE A5 FC
B5F0: 18 69 02 90 02 E6 FD 85 FC
B5F9: A4 EE
B5FB: 60
```

1.2.4. *octave- and tempo-switching in real time.* As with spookbit, we can lower the pitch range quite simply by modifying select dex/bne and dey/bne blocks into jumps. The addresses involved are easy to figure out.

Tempo-switching is easier than in spookbit, somehow, as the rejiggered code now always loads a specific constant into the X or Y register for every pulse width unit. Modifying the byte `$7F` at `$B501` and `$B58F` is sufficient.

1.3. **note-to-hex lookup table.** The two engines share the same pitches for given pulse widths. Here A4 = 436 Hz, mostly to mitigate errors in D6.

A2	(255) ($FF)	A#2 (Bb2)	246 $F6	B2	232 $E8	C3	219 $DB	C#3 (Db3)	206 $CE	D3	195 $C3
D#3 (Eb3)	184 $B8	E3	174 $AE	F3	164 $A4	F#3 (Gb3)	155 $9B	G3	146 $92	G#3 (Ab3)	138 $8A
A3	130 $82	A#3 (Bb3)	123 $7B	B3	116 $74	C4	109 $6D	C#4 (Db4)	103 $67	D4	97 $61
D#4 (Eb4)	92 $5C	E4	87 $57	F4	82 $52	F#4 (Gb4)	77 $4D	G4	73 $49	G#4 (Ab4)	69 $45
A4	65 $41	A#4 (Bb4)	61 $3D	B4	58 $3A	C5	54 $36	C#5 (Db5)	51 $33	D5	48 $30
D#5 (Eb5)	46 $2E	E5	43 $2B	F5	41 $29	F#5 (Gb5)	38 $26	G5	36 $24	G#5 (Ab5)	34 $22
A5	32 $20	A#5 (Bb5)	30 $1E	B5	29 $1D	C6	27 $1B	C#6 (Db6)	25 $19	D6	24 $18

2. ANIMATION

The last shot of the UnderTale intro is a vertical pan. This is ridiculously difficult to do in an Apple II with a full image, where the deltas are non-trivial and the display lines are not contiguous in memory.

The `SCROLLER` routine handles the pan via a lookup table and swapping between the two hi-res pages. In addition to scrolling down whatever is on the screen, the routine also scrolls in image data just beyond the hi-res pages, and this part of the lookup table has to be continuously updated.

```
B000:  18 f0 c8 a0 78 50 28 00
B008:  00 00 00 00 00 00 00 00 80 80 80 80 80 80 80 80
B018:  00 00 00 00 00 00 00 00 80 80 80 80 80 80 80 80
B028:  00 00 00 00 00 00 00 00 80 80 80 80 80 80 80 80
B038:  00 00 00 00 00 00 00 00 80 80 80 80 80 80 80 80
B048:  28 28 28 28 28 28 28 28 a8 a8 a8 a8 a8 a8 a8 a8
B058:  28 28 28 28 28 28 28 28 a8 a8 a8 a8 a8 a8 a8 a8
B068:  28 28 28 28 28 28 28 28 a8 a8 a8 a8 a8 a8 a8 a8
B078:  28 28 28 28 28 28 28 28 a8 a8 a8 a8 a8 a8 a8 a8
B088:  50 50 50 50 50 50 50 50 d0 d0 d0 d0 d0 d0 d0 d0
B098:  50 50 50 50 50 50 50 50 d0 d0 d0 d0 d0 d0 d0 d0
B0A8:  50 50 50 50 50 50 50 50 d0 d0 d0 d0 d0 d0 d0 d0
B0B8:  50 50 50 50 50 50 50 50 d0 d0 d0 d0 d0 d0 d0 d0
B100:  61 60 60 60 60 60 60 60
B108:  20 24 28 2c 30 34 38 3c 20 24 28 2c 30 34 38 3c
B118:  21 25 29 2d 31 35 39 3d 21 25 29 2d 31 35 39 3d
B128:  22 26 2a 2e 32 36 3a 3e 22 26 2a 2e 32 36 3a 3e
B138:  23 27 2b 2f 33 37 3b 3f 23 27 2b 2f 33 37 3b 3f
B148:  20 24 28 2c 30 34 38 3c 20 24 28 2c 30 34 38 3c
B158:  21 25 29 2d 31 35 39 3d 21 25 29 2d 31 35 39 3d
B168:  22 26 2a 2e 32 36 3a 3e 22 26 2a 2e 32 36 3a 3e
B178:  23 27 2b 2f 33 37 3b 3f 23 27 2b 2f 33 37 3b 3f
B188:  20 24 28 2c 30 34 38 3c 20 24 28 2c 30 34 38 3c
B198:  21 25 29 2d 31 35 39 3d 21 25 29 2d 31 35 39 3d
B1A8:  22 26 2a 2e 32 36 3a 3e 22 26 2a 2e 32 36 3a 3e
B1B8:  23 27 2b 2f 33 37 3b 3f 23 27 2b 2f 33 37 3b 3f
B200:  D8           CLD
B201:  A0 9D        LDY #$9D
B203:  F0 36        BEQ +$34
B205:  98           TYA
B206:  18           CLC
B207:  69 05        ADC #$05
B209:  A8           TAY
B20A:  B9 02 B0     LDA $B002,Y
B20D:  85 42        STA $42
B20F:  B9 02 B1     LDA $B102,Y
B212:  49 60        EOR #$60
B214:  85 43        STA $43
B216:  98           TYA
B217:  38           SEC
B218:  E9 06        SBC #$06
```

```
B21A:  A8          TAY
B21B:  B9 02 B0    LDA $B002,Y
B21E:  BE 02 B1    LDX $B102,Y
B221:  85 3C       STA $3C
B223:  86 3D       STX $3D
B225:  18          CLC
B226:  69 27       ADC #$27
B228:  85 3E       STA $3E
B22A:  90 01       BCC +$01
B22C:  E8          INX
B22D:  86 3F       STX $3F
B22F:  84 CF       STY $CF
B231:  A0 00       LDY #0
B233:  20 2C FE    JSR $FE2C
B236:  A4 CF       LDY $CF
B238:  4C 03 B2    JMP $B203 [back to BNE]
B23B:  A2 08       LDX #$08
B23D:  CA          DEX
B23E:  30 10       BMI [forward to RTS]
B240:  BD 00 B0    LDA $B000,X
B243:  18          CLC
B244:  69 F0       ADC #$F0
B246:  9D 00 B0    STA $B000,X
B249:  90 F2       BCC [back to DEX]
B24B:  FE 00 B1    INC $B100,X
B24E:  B0 ED       BCS [back to DEX]
B250:  8D 55 C0    STA $C055 [switch to page 1/2 if 54/55]
B253:  A2 C9       LDX #$C9
B255:  CA          DEX
B256:  F0 0B       BEQ [forward towards RTS]
B258:  BD 07 B1    LDA $B107,X
B25B:  49 60       EOR #$60
B25D:  9D 07 B1    STA $B107,X
B260:  4C 55 B2    JMP $B255 [back to DEX]
B263:  AD 51 B2    LDA $B251
B266:  49 01       EOR #$01
B268:  8D 51 B2    STA $B251
B26B:  60          RTS
```