This lab presents two devices, both partially digital, that have in common the use of feedback to generate an output related in a useful way to an input signal. The first circuit, an analog-to-digital converter (“ADC”), uses

---

### Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>18L.1 Analog to Digital Converter (“ADC”)</td>
<td>2</td>
</tr>
<tr>
<td>18L.1.1 D/A Converter (“DAC”)</td>
<td>2</td>
</tr>
<tr>
<td>18L.1.2 ADC: Watching the Conversion Process</td>
<td>3</td>
</tr>
<tr>
<td>18L.1.3 Speed Limit</td>
<td>6</td>
</tr>
<tr>
<td>18L.1.4 Completing the ADC: Latching the Digital Output</td>
<td>6</td>
</tr>
<tr>
<td>18L.2 Phase-Locked Loop: Frequency Multiplier (We hope you’ll build this; but some people will not have time.)</td>
<td>7</td>
</tr>
<tr>
<td>18L.2.1 Generating a multiple of line frequency: (Type II detector)</td>
<td>8</td>
</tr>
<tr>
<td>18L.2.2 Try FM demodulation (in slow motion)</td>
<td>10</td>
</tr>
<tr>
<td>18L.2.3 Expanded Lock Range: × 64 rather than × 1k</td>
<td>11</td>
</tr>
</tbody>
</table>

---

1Revisions: add header file and index (7/14); slightly revise pll figure, spreading it out (3/12); fix pin mention for Q6, p. 11, add footnote saying that National’s pin numbering begins with Q1, strangely (2/12).
feedback to generate the digital equivalent to an analog input voltage. The second circuit, a phase-locked loop ("PLL"), uses feedback to generate a signal matched in frequency to the input signal—or to some multiple of that frequency.

18L.1 Analog to Digital Converter ("ADC")

The A/D conversion method used in this lab, “successive approximation,” or “binary search,” is probably the method still most widely used, though now competing with delta-sigma designs. It provides a good compromise between speed and low cost. It substitutes some cleverness for the “brute force” (that is, large amounts of analog circuitry) used in the fastest method, flash or parallel conversion.²

Note that the converter you will build today with four chips normally would be fabricated on a single chip. We build it up, using an essentially obsolete successive-approximation register (SAR), so that you will be able to watch the conversion process. In an integrated converter the approximation process is harder to observe because the successive analog estimates are not brought out to any pin, though the stream of bits that forms that estimate may be. We also omit, for simplicity, the sample-and-hold that normally is included.

![Figure 1: Successive-approximation a/d converter: block diagram](image)

18L.1.1 D/A Converter ("DAC")

The process of converting digital to analog is easier and less interesting than converting in the other direction. In the successive-approximation ADC, as in any closed-loop ADC method, a DAC is necessary to complete the feedback loop: it provides the analog translation of the digital estimate, allowing correction and improvement of that digital estimate. As a first step in construction of the ADC we will wire up a DAC.

This DAC, the Analog Devices AD558, integrates on one chip not only the DAC, but also an output amplifier and an input latch. The latch is of the transparent rather than edge-triggered type, and we will ignore it in this lab, holding the latch in its transparent mode throughout (grounding pins 9 and 10, CS* and CE*, keeps the latch continuously transparent).

²Yes, we admit, it is an unusual brute who could put together a flash ADC.
Checkout:

(Hurry through this checkout. The fun comes later!)

Confirm that the 558 is working, by controlling its two MSB’s and its LSB with a DIP switch or simply with wires plugged into ground or +5. Hold the other five input lines low. (You may find a ribbon cable convenient, though less tidy than a wire-at-a-time; you will need to feed all 8 lines in a few minutes, in any case.)

Note the relation between switch settings and $V_{out}$. Full-scale $V_{out}$ range should be about 0 to 3.8 v. What “weight” (in output voltage swing) should $D_7$ carry? $D_6$? $D_5$?

Note the weight of the LSB. Here, if you look at the output with a scope you are likely to find noise of an amplitude comparable to that of the signal. If the noise is at a very high frequency, however, it may be quite harmless. How does your DVM appear to treat this noise? How do you expect the comparator to treat it when you insert the DAC into the ADC loop?

Digital Input:

<table>
<thead>
<tr>
<th>D7</th>
<th>D6</th>
<th>D5...D1</th>
<th>D0</th>
<th>$V_{out}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure 3: DAC Checkout: voltage weights for particular input bits

18L.1.2 ADC: Watching the Conversion Process

When you are satisfied that the DAC is working, add the rest of the successive-approximation converter circuit: comparator and SAR along with DAC. If you are using a 74LS503 rather than 74LS502, you must ground pin 1, an ENABLE\(^*\) pin; we do not use pin 1 in the ‘502 circuit,\(^5\) and can leave it open if using a ‘502.

If you have neither ‘503 nor ‘502 but are using a custom-programmed PAL that emulates the ‘502 please note

---

\(^3\)Yes, $D_7$ should carry the weight of one half full scale; $D_6$, one quarter.

\(^4\)We mean to suggest that very high-frequency noise—radio frequency that one often sees when scope gain is very high—may be too quick to be sensed by this ADC circuit, just as it is too quick to make the DVM respond. Large high-frequency signals, however, can be accidentally rectified by a DVM, and can produce spurious results, as you may possibly have seen back in the nasty oscillators lab, when the discrete follower sometimes can make a DVM claim to see a voltage beyond the positive supply.

\(^5\)On the ‘502 it serves as a synchronized version of the D input; not useful for our purposes.
that its pinout will not match that of the other two chips. Consult its own datasheet.

![Diagram of circuit](image)

**Figure 4: Successive-approximation a/d converter: slow-motion checkout: layout**

Connect the 8 DAC inputs to the 8 breadboard LEDs, so that you can watch the estimating process.

### 18L.1.2.1 Slow Motion Checkout

Use a debounced switch for Clock and another switch (which may bounce) for Start* (S*). Watch Conversion Complete* (CC*) on an LED.

- You will have to limit the current to this 9th LED, with a resistor. To choose the resistor value, consider the following points: an LS-TTL output can sink 8mA, plenty for a high-efficiency LED. The LED drops about 2V when lit.

*Note*: the behavior of S* may strike you as odd:

- First, the SAR is “fully synchronous;” asserting S* by itself has no effect. The SAR ignores S* unless you clock the SAR while asserting S*.
- Second, “Start*” is poorly named. It should be called something like “initialize*,” because conversion will not proceed beyond the initial guess until you release S*.

Ground the analog input\(^{6}\) and walk the device through a conversion cycle in slow motion. Watch the digital estimates on the 8 LED’s and the analog equivalents, the analog estimates, on the DVM or scope.

As you clock slowly through a conversion cycle, the DAC output (showing the analog equivalent of the SAR’s digital estimates) should home in on the correct answer, 0 volts. Do you recognize the binary search pattern in these successive estimates? If you get a digital 1 rather than 0 as your converter’s final, best estimate—that is, if the LSB ends up HIGH rather than LOW—make sure that you have grounded the input close to the converter. Then use the scope to look closely at ground and +5V lines, watching for noise. A digital . . . 01 outcome need not shock you, given that an LSB is only about 15mV, and the comparator’s \(V_{\text{OFFSET}}\) is spec’d at 5mV (max); a small drop in the ground line added to this \(V_{\text{OS}}\) could tip the LSB.

### 18L.1.2.2 Operation at Normal Speed

Now make three changes:

\(^{6}\)Note that the “analog input” is not the non-inverting terminal of the comparator, but one end of the 10k resistor; that resistor is necessary to maintain hysteresis.
1. Connect “Conversion Complete***” (CC*) to “Start***” (S*). This lets the converter tell itself to start a new conversion cycle as soon as it finishes carrying out a conversion. (Disconnect the pushbutton that was driving S*, of course.)

![Figure 5: CC* wired to drive start*: SAR starts self](image)

2. Feed the converter from a pot (2.5k or less: (Why?)?), rather than from ground:

![Figure 6: Variable DC analog input](image)

3. Clock the converter with a TTL-output oscillator, rather than with the pushbutton. (The oscillator built into your breadboard is convenient.) Let $f_{\text{clock}} \approx 100\text{kHz}$.

Watch the DAC output and ADC input on the scope. (If you want a rock-steady picture, trigger the scope on CC*.) Vary the pot setting (analog input voltage), and confirm that the converter homes in on the input value.

18L.1.2.3 Displaying Full Search “Tree”

You have watched the converter put together its best estimate, homing on the input value. If you feed the converter all possible input values, you can get a pleasing display that shows the converter trying out every branch of its estimate “tree.”

![Figure 7: Displaying binary search “tree”](image)

Feed the converter an analog input signal from an external function generator: a triangle wave spanning the converter’s full input range (0 to about 4V). Set the frequency of the triangle wave at about 100 Hz. Trigger the scope on CC*.

If necessary, tinker with the frequency and amplitude of the input waveform, until you achieve a display of the entire binary search tree. This can be a lovely display (you may even notice the “quaking aspen” effect). You are privileged to see the binary search in such vivid form—while it remains to most other people only an abstraction of computer science. (An exceptionally noise-free image of the search tree produced by this lab circuit appears as AoE’s fig. 13.16.)

---

7You will recognize, here, our usual concern that $R_{\text{source}}$ not mess things up. Here, an over-large pot would mess things up by causing a significant increase in hysteresis.
18L.1.3 Speed Limit

The ADC completes an 8-bit conversion in nine clock cycles. Evidently, the faster you can clock it, the faster it can convert. The faster it can convert, the higher the frequency of the input waveform that you can capture (you need a shade more than two samples per period, in theory; in practice you may want to take 3 to 5 samples).

How fast can you clock your converter? Here’s what must happen between clock edges:

![Figure 8: ADC speed limit: what must be accomplished within one clock period](image)

These numbers suggest a maximum clock frequency of a little less than 600kHz.

Feed the converter a DC level and gradually crank up the clock rate as you watch the analog estimates on the scope. This time, use the external function generator as clock source (the breadboard’s top frequency, 100kHz, is too low). At some clock frequency you will recognize breakdown: the final estimate will change, because the clock period will no longer be allowing time for all levels to settle. Chances are, this will happen at a frequency well above the worst-case value of 600kHz (above 1MHz, in most cases we have seen.)

18L.1.4 Completing the ADC: Latching the Digital Output

Up to this point we have been looking at the converter’s feedback DAC output. Do not let our attention to this analog signal distract you from the perhaps-obvious fact that the feedback-DAC output is not the converter’s output. We use an ADC in order to get a digital output, of course, and on a practical IC ADC, as we observed at the start of these lab notes, the analog estimate is not even brought out to any pin.

We now return our attention to the normal subject of interest: the ADC’s digital output.

**Output Register**

An integrated ADC normally includes a register to save its output (and incidentally, in the age of microcomputers, such a register routinely includes 3-state outputs, for easy connection to a computer’s data bus).

Now we will add an 8-bit register of D flip-flops to complete the ADC. We need to provide a clock pulse, properly timed, that will catch the converter’s best estimate and hold it till the next one is ready. Timing concerns make this task more delicate than it looks at first glance.

CC* ("Conversion Complete") certainly sounds like the right signal. It turns out that it is not—not quite. The trailing edge (rise) of CC* comes too late; the other edge (which, inverted, could provide a rising edge) comes too early.

At the beginning (fall) of CC*, the SAR is putting out its initial estimate for the LSB; it has not yet corrected it (set it high), if such correction is necessary. Thus you would lose the LSB data, getting a constant Low, if you somehow used the start of CC* to latch the output.

At the end (rise) of CC*, the SAR is already presenting the first guess of its next cycle (0111 1111).\(^8\)

---

\(^8\)Why? Because the rise of CC* and the initial guess both come in response to the SAR clock, and CC* happens to come up a little later (see 74LS502 data sheet: t\(_{PLH}\) is slower than t\(_{PHL}\). How’s that for fine print?)
What we need is a pulse that ends well away from both edges of CC*:

A single gate can do the job.

Add this gate and feed its output to the clock of the output register (74HC574). Let the register’s outputs drive the breadboard’s eight LEDs. (Don’t forget power and ground, not shown in fig. 11. They are the corner pins, as usual for digital devices.)

Watch the converter’s digital output and confirm that it follows the analog input applied from the potentiometer. It may be indecisive, by 1 bit. Is this indecision avoidable?

18L.2 Phase-Locked Loop: Frequency Multiplier (We hope you’ll build this; but some people will not have time.)

You will have a chance to apply this PLL, if you like, in Lab Micro 4— as we mentioned in today’s class notes. In that lab we will need to generate a multiple of the frequency at which the computer samples a waveform (we want a square wave at $32 \times$ the sampling rate); we will use that multiple to regulate $f_{3\text{dB}}$ of a low-pass filter. This adjustable filter is of the switched-capacitor type, like the one you built in the MOSFET lab (assuming you got that far).

9We have nothing subtle in mind; we’re just referring, as in §18L.1.1 on page 2, to the likelihood that noise may be comparable to the value of an LSB, in your breadboarded circuit.
Thus we will make the filter follow our sampling rate (so as to clear away the spurious high-frequency elements in the DAC’s steppy output waveform). This future application for the PLL explains some elements of its design: particularly, its frequency range, set to cover most of the range of sampling rates you’re likely to adopt.

We will first apply the loop, however, as if we were using it to generate a multiple of the line frequency, 60Hz. This example is discussed in detail in AoE §13.3.4, and the circuit below is the one designed in that discussion, except that we have altered the VCO component values to permit a wider range of operation.

![Phase locked loop frequency multiplier circuit (using 74HC4046 PLL)](image)

Construct the circuit shown in the figure above. The phase detector and VCO are drawn as separate blocks, above; but note that they are within one 4046 chip: you do not need two 4046s. The 4040 is a 12-stage ripple counter.

18L.2.1 Generating a multiple of line frequency: (Type II detector)

Take the replica signal from $Q_9$ of the 4040 (pin 14) ($Q_9$ divides $f_{\text{clock}}$ by $2^{10}$). In case this seems odd, recall that $Q_0$ divides by two, not one.) Set the function generator, which drives the input, to around 60Hz. Use the scope to compare this input signal with the synthesized replica at 4046 pin 3. Confirm that the PLL locks onto the input frequency within a few seconds. Are the two waveforms in phase? The lock LED should light when the loop is locked (a logic high at pin 1 indicates that the phase detector output is 3-stated: that is, the detector is satisfied, rarely seeing reason to correct the replica frequency).

See if the replica follows the input as you change frequency slowly; then try teasing the PLL by changing frequency abruptly. You should be able to see a brief hunting process before the loop locks again.

The limit of input frequency capture and lock ranges (identical, for this PLL) is determined by the VCO.
range: from about 40kHz to 250kHz, for the specified values of $R_1$, $C_1$: 220k, 220pF. Since the counter in the feedback path divides by $2^{10}$, the input frequency range runs from approximately 40Hz to 250Hz.

18L.2.1.1 The Loop Filter

The hard part of PLL design comes in designing the low-pass filter to maintain stability. The PLL uses a feedback loop strikingly reminiscent of the loop in the PID motor-control lab: in the PID exercise, the motor-to-position transducer placed an integration inside the loop. This $90^\circ$ lag obliged us to use care to avoid injecting another $90$-degree lag, at frequencies where loop gain was above unity.

In the PLL, again an integration lies within the loop, and again it threatens us with instability. This time, the integration results from the fact that we are sensing a characteristic (phase) that is an integral of the characteristic that our loop means to control (frequency). Here, the integration is harder to localize than in the PID loop, where it clearly occurred in the motor-potentiometer unit. In the PLL the integration—the equivalent of the motor-pot combination—is performed by the VCO-phase-detector pair. The detector’s output is a signal that measures phase error.\footnote{You may be inclined to ask, “Why do this? Why not be more straightforward and measure frequency error directly?” The answer is that we have no way to do that quickly. To measure frequency even indirectly, inferring it from period, requires waiting for a full period of the two waveforms. Phase error information is available more promptly: after each of the respective rising edges, in the Type II detector.}

For a constant frequency difference, the phase error (difference between input signal and the replica generated by the VCO) ramps; in other words, it integrates the frequency error. You will find a scope image that demonstrates this ramping effect in today’s class notes. This behavior closely resembles what we saw in the motor-pot combination: a constant error signal fed to the motor-pot produced a ramping output voltage: integration, once more.\footnote{If you’re not exhausted by this discussion, you may want to consider a complication: a second integration performed by the $RC$, at least as used in the edge-sensitive “Type II” detector: a constant phase difference produces a ramping output from the phase detector. This second integration is analogous to the $I$ used in the PID circuit. The $I$ term drives the PID error to zero; the Type II detector, similarly, uses this integration to drive the phase difference to zero.}

We need a low-pass filter in the loop, to smooth the signal into the VCO—but we cannot afford another $90^\circ$ lag. What is to be done? The answer turns out to be a simple circuit amendment: use a low-pass whose phase shift at high frequencies goes toward zero rather than toward $-90^\circ$. A low-pass with an extra resistor between cap and ground does the trick, as today’s class notes argue.

If you replace the filter’s 330k resistor with 22k, you should find that the loop hunts much longer—overshooting, backtracking…. Watching the demod signal, at pin 10, which shows the VCO input voltage, may make the hunting and overshooting vivid for you.

By reducing that $R$ value, you have made a dangerous reduction in the phase margin—the safety margin between the phase shift around the loop (at the frequency where the loop gain goes to unity) and the edge of the cliff: the deadly $-180^\circ$ shift. This is a notion you will recall from Lab 9 (“…Nasty Oscillators”) and from the PID lab.

If you want to live really dangerously, short out the lower resistor; now you have reduced the phase margin to zero, and the circuit may hunt forever. When you have seen this effect, restore the 330k, to restore stability.

What frequency should be present at the VCO output when the loop is locked? Check your prediction by looking with the scope at that point (pin 4 of the 4046). Why is this waveform jittery? (in a world without noise it would not be.)\footnote{The jitter arises because the corrections to VCO frequency come only once in every 1024 cycles of the VCO output. Slight wanderings of frequency, between these checks, go uncorrected. As usual, being college teachers, we are reminded of the fact—probably beneficial to humans, though unsettling in the PLL analogy—that the college checks whether students can do the course work only now and then: at best, weekly; the big test comes only once per term. In the meantime, who knows what goes on in the student’s life? No doubt that life would be full of jitter, if one were so rude as to look at it between checkpoints.}

Now look at the output of the phase detector (pin 13 of the 4046). This is the type II detector described in Ch. 9 (sec. 9.27, pp. 644-45). You’ll notice a string of brief positive going pulses, each decaying away
exponentially when the detector reverts to its 3-stated condition (you may also see smaller negative-going pulses, to the extent that the circuit is troubled by noise).

Theory predicts that these correction pulses should vanish in the steady state. But the 10 megohm load of the scope probe you are using is discharging the filter capacitor fast enough to cause the pulses; the current the scope draws also causes the VCO to be slightly under-driven, causing the VCO to run a little late; a lagging phase difference persists: the loop now needs a phase difference, so as to provide the positive pulses that feed the scope's $R$ to ground. If you're feeling energetic, interpose a 358 op amp follower between the capacitor and the probe. The positive pulses should become narrower—and now the phase difference is reversed: the VCO output comes a little early.\textsuperscript{14}

\section*{18L.2.2 Try FM demodulation (in slow motion)}

If you watch the input to the VCO (which is available in buffered form at pin 10), you can see a measure of VCO frequency. That’s not very interesting when the input frequency is constant—but becomes interesting when a variation in input frequency carries information, as it does in an FM radio broadcast (or in our analog project, where we sent music across the room with an LED’s flashes). We aren’t going to pause long enough to do such an exercise, here; but we suggest that you get a glimpse of the way it might work, with a two-minute demonstration.

Replace the $1\mu F$ capacitor, temporarily, with a $0.1\mu F$ part (to make the loop adjust faster), and watch the demod signal (pin 10) as you try to vary $f_{IN}$ sinusoidally, by slowly varying the function generator frequency. You should sweep the scope very slowly—using “roll” mode (perhaps 1s/div) if you have a digital scope. We hope the demod signal will look like the sinusoid that varies or “modulates” the PLL’s input frequency. Too bad we didn’t know about PLL’s when we did the group audio-LED project.

When you’ve had enough fun with this, restore the $1\mu F$ capacitor.

\textbf{Type I Detector (exclusive-OR)}

The 74HC4046 includes three phase detectors. AoE describes two of them (types I and II) in §13.3.2.1.\textsuperscript{15} The third detector on the HC4046, Type III, is simply a variation on the SR latch, with $S$ and $R$ driven by positive levels on the input and replica lines, respectively. We will ignore the Type III detector;\textsuperscript{16} two detectors seem plenty to consider on a first encounter with a PLL. But feel free to check out the Type III, if you like: its output appears at pin 15.

The Type I detector output is at pin 2, and the inputs are the same as for the type II detector; to use the Type I, simply move the wire from pin 13 to pin 2 (and then to 15 for the Type III, if you must!). For both Type I and Type III you should be able to see the fluctuation of the VCO frequency over the period of the input, which you can exaggerate by reducing the size of the $1\mu F$ loop filter capacitor.

If you make a sudden, large change in the input frequency, you should be able to fool the type I circuit into locking onto a harmonic of the input frequency (a multiple of the input frequency). For our purpose in Lab Micro 4, such an error would make the circuit useless, so we will use the Type II detector. You are likely to make the same choice in most applications.

Note also the phase difference that persists between input and feedback signal in the locked state. This simple phase detector (like the Type III) requires such a phase difference; this difference generates the signal that drives the VCO. If the phase difference ever goes to zero (or to $\pi$), the loop loses feedback: it can no longer correct frequency in both senses, as required. When it hits that limit, it’s like an op amp feedback loop

\footnote{Why? It's a fussy detail, but now, instead of discharging, as the scope probe did, the '358's $I_{BIAS}$ injects some current from its PNP input transistor's base.} \footnote{AoE's discussion treats the similar CD4046 rather than the 74HC version.} \footnote{The type III detector was added as a sort of afterthought; early versions of the 4046 did not offer it. Like the simple XOR (Type I) it requires a phase difference between input and replica signal.}
that fails because the op amp output hits saturation and thus cannot make a further correction. The Type II detector is altogether classier: it requires no errors to keep the loop locked; it is able to use the capacitor as a sample-and-hold, once the loop is locked, rather than as a conventional filter.

When you have finished looking at the behavior (and misbehavior) of the Type I detector, revert to the earlier circuit, using the Type II (output at pin 13).

### 18L.2.3 Expanded Lock Range: × 64 rather than × 1k

Now let’s set up the PLL as you will want to find it the next time you use it, in Lab Micro 4: change the tap on the 4040 from Q₉ to Q₅ (pin 2). Now you are generating, at the VCO output, a modest 64× fᵢ₉. Because we are feeding back a larger part of the output frequency, we need to attenuate more, in the loop filter. (We are worrying about the loop gain, as we did for op amps.)

Feeding back 1 part in 64, rather than 1 in 1024—about 16× as much—calls for cutting the fraction preserved by the loop filter proportionately. So, we cut the 330k that was below the 1µF cap by a factor of 16, to about 22k. The two sets of dividers—analog filter and digital counter—are sketched below, for the two stages: stage one, where the loop multiplies by 1024, and stage two, where the loop multiplies by 64. The fraction fed back

![Diagram of PLL filter divider](image)

Figure 13: PLL filter divider is adjusted to hold fraction fed back roughly constant in the two stages

You may notice that we have drawn the “loop filter” as if it were simply a resistive divider. We have done that because at the high frequencies where we anticipate challenges to stability Xₐ is insignificant relative to the R values.

Over what range of input frequencies does the PLL now remain locked? The range should be wide; we need this range in order to make the sampling scheme of Lab Micro 4 flexible. The 4046 is capable of capture and lock over a frequency range of about 6:1.¹⁷ We will be content with a range that accepts an input between about 600Hz and 4kHz. Capture and lock range are the same, for the Type II detector; for the less clever detectors, capture range (the range over which the loop will be able to achieve lock) is narrower than lock range (the range over which the loop will hang on once it has locked). This ability of the Type II to capture any frequency it can hold seems to follow from its immunity to harmonics: you can’t fool the Type II.

¹⁷This is the approximate range as one drives the VCO over its permitted range from 1.1V to 3.4V (the data sheet specifies this range with a 4.5V supply; at 5V each voltage is presumably about 10% higher).
Index

74HC4040 (lab), 8
74HC4046 (lab), 8
74HC574 (lab), 6
74LS503 (lab), 3
AD558 (lab), 2
ADC (lab), 3–7
binary search (lab), 2
binary search display (lab), 5
CC* (lab), 4
DAC (lab), 2–3
FM demodulation
   PLL (lab), 10
phase-locked loop
   FM demodulation (lab), 10
   loop filter (lab), 9
   phase detector
   XOR (lab), 10
phase-locked loop (lab), 7–11
SAR (lab), 2