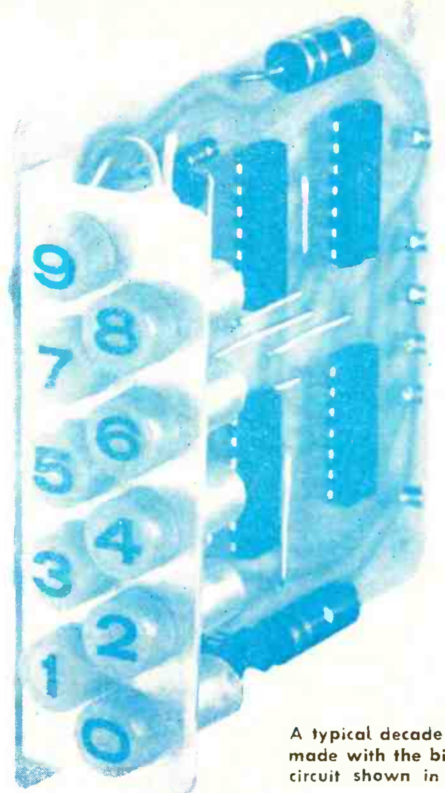
IC Decimal Counting Techniques

By DONALD E. LANCASTER

IC's have made complex digital counter circuits simpler and less expensive than those made with discrete or mechanical components. Here are several practical schemes which can be used to time events.



A typical decade counter made with the biquinary circuit shown in Fig. 8.

RECENTLY, we've seen how digital integrated circuits have opened the doors to a wide variety of new applications. One of the most important of these new uses is decimal counting and readout circuits; typically, event and frequency counters, digital voltmeters and multimeters, electronic clocks, calculators, computers, and so forth. IC's have made these circuit designs simpler and lower in cost than older tube and solid-state versions, and significantly faster, smaller, and more reliable than mechanical or gas-tube counting techniques.

A basic problem common to all these applications is: to *count* an input signal by one's up to nine; *decode* each counter state; *indicate* each digit; *carry* a pulse to the next stage every time there is a count of 9 + 1; and, finally, provide a *reset* capability that allows the counter to be put into the "0" state at the beginning of a measurement.

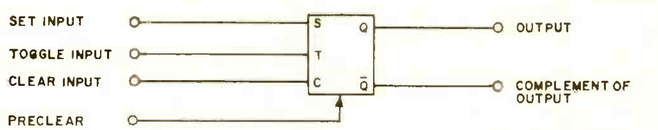
We do not have an integrated circuit that will single-handedly do all these tasks, while simultaneously lighting up a large numeral on its case. We must make use of several integrated circuits in combination with a conventional readout. But, depending upon the readout and the speed with which we must operate, the cost, legibility, and visibility factors, we have a wide choice of decade counting schemes. Let's look at some of the more popular techniques, and see just how they may be put to use.

The Building Block: The JK Flip-Flop

The basic building block of all counting systems is some device that has *two* possible states, "on" and "off". With few exceptions, the basic building block is a JK flip-flop, a circuit that is somewhat similar to the older types of flip-flops (self-steering bistable multivibrators or Eccles-Jordan circuits), but far more versatile and lower in cost. Several of these flip-flops can be tied together to achieve a system capable of many states. For example, if we connect them as divide-by-two circuits so that the outputs run 1, 2, 4, 8, 16, 32. . . . we obtain a binary divider chain or binary counter. By modifying the count procedure, we can generate several different decimal counting techniques. Or, we might choose to connect the flip-flops in a different manner, perhaps as a "bucket brigade" lining them up so that they pass on a count under command. This is called a shift register and if we connect the output of the register back to its own input, we have a ring counter and several more possibilities for decimal counting.

Fig. 1 gives us a closer look at a typical JK flip-flop. Notice

Fig. 1. (A) JK flip-flops form basic building block of many decimal counters. The table (B) shows counter state at the instant of toggle. Diagrams in (C) and (D) show the proper connections for a binary divider and for shift register.



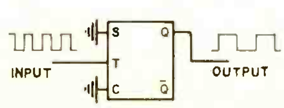
NOTE: EACH FLIP-FLOP ALSO REQUIRES B+ AND ⏚. THESE CONNECTIONS ARE USUALLY NOT SHOWN ON LOGIC DIAGRAMS

(A)

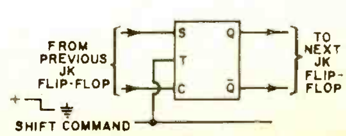
TABLE APPLIES TO: µL923 MC723P 1/2 MC791 RTL TYPES, ETC.

IF THE SET INPUT IS	AND THE CLEAR INPUT IS	AND THE TOGGLE IS	THE OUTPUT WILL
⏚	⏚	+ → ⏚	CHANGE TO THE OTHER STATE
⏚	+	+ → ⏚	GO TO THE ⏚ STATE
+	⏚	+ → ⏚	GO TO THE + STATE
+	+	+ → ⏚	DO NOT CHANGE STATE

(B)



(C)



(D)

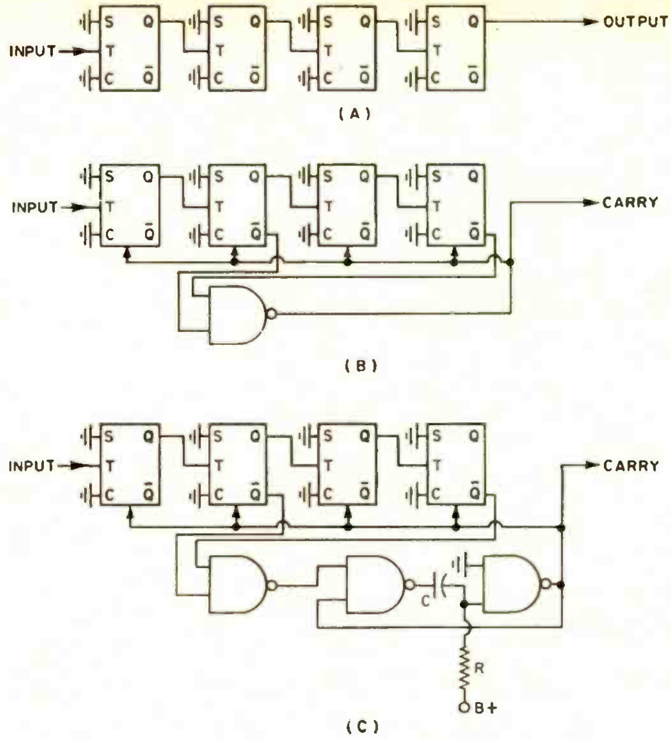


Fig. 2. (A) Four flip-flops count 0 to 15 in a 1-2-4-8 binary divider chain. A "nand" gate (B) can be used to detect state "10" and preclear counter to "0". One method of eliminating preclear difficulty is to use circuit connections as shown in (C).

that there are two outputs, the Q output and the Q output. These outputs are complementary because one output is "+" and the other is at ground. Under certain combinations of inputs, the output can be made to change state, with the positive output going to ground, and the grounded output going positive.

There are two groups of inputs, the Set, Toggle, and Clear inputs and, off in a class by itself, the Preclear input.

The Preclear input is normally grounded. If the Preclear input were made positive, the JK flip-flop would immediately go into the state in which the Q output is grounded and the Q output is "+", if it were not there already. This happens regardless of the status of the other inputs. Thus, the Preclear input must be used to "empty" a counter or register, making the readout indicate zero.

The other three inputs are always used together and, depending upon their input voltages, can make the JK flip-flop change state, go into one of the two possible output states, or do nothing. The key to the operation is the Toggle input. The flip-flop cannot change state except during the instant the Toggle input abruptly changes from a positive to a grounded condition. To operate the flip-flop, we first apply voltages to the Set and Clear inputs (this is called conditioning) and then we toggle the flip-flop by causing the Toggle input to suddenly change from a positive to a grounded condition.

Specifically, a JK flip-flop made with RTL digital IC's obeys the following laws: if the Set input is grounded and the Clear input is grounded, the flip-flop will change to the other state upon a negative-going Toggle transition; if the Set input is "+" and the Clear input is grounded, the flip-flop will go into the state in which the Q output is "+" upon a negative-going Toggle transition; if the Set input is grounded and the Clear input is made "+", the flip-flop will go into the state in which the Q output is grounded upon a negative-going Toggle transition; if the Set input is made "+" and the Clear input is made "+", nothing happens when a negative-going Toggle transition occurs.

Thus, to design a binary divider, we simply ground the Set and Clear inputs (Fig. 1C). Note that the output goes from "+" to ground only on every other negative toggle transition (in between, it goes from ground to "+") and we have a divide-by-two circuit. The next stage divides by two again so it only responds to every second Toggle negative transition, and its output has a negative transition only once every four Toggle transitions. The next stage divides by eight; the next by sixteen; and so on. By observing the state of each binary divider, we can see how many Toggle negative transitions have taken place since we last Precleared the counter. In practice, we feed input pulses to the binary divider and the fall time of each input pulse toggles the divider chain. Thus we can easily count the number of input pulses by simply observing the state of the binary divider.

Another way to use the JK flip-flop is shown in Fig. 1D. Here we have built a shift register by connecting each JK output directly to the Set and Clear inputs of the next flip-flop. All toggle inputs are connected in parallel and operated synchronously. Upon every negative toggle transition, the state of each flip-flop is passed on to the next and so on down the line. An automatic time-delay circuit is built into each JK flip-flop to prevent the states from being transferred more than one stage each command.

Together, these two basic JK flip-flop interconnections form the basis for the majority of our decimal counting methods.

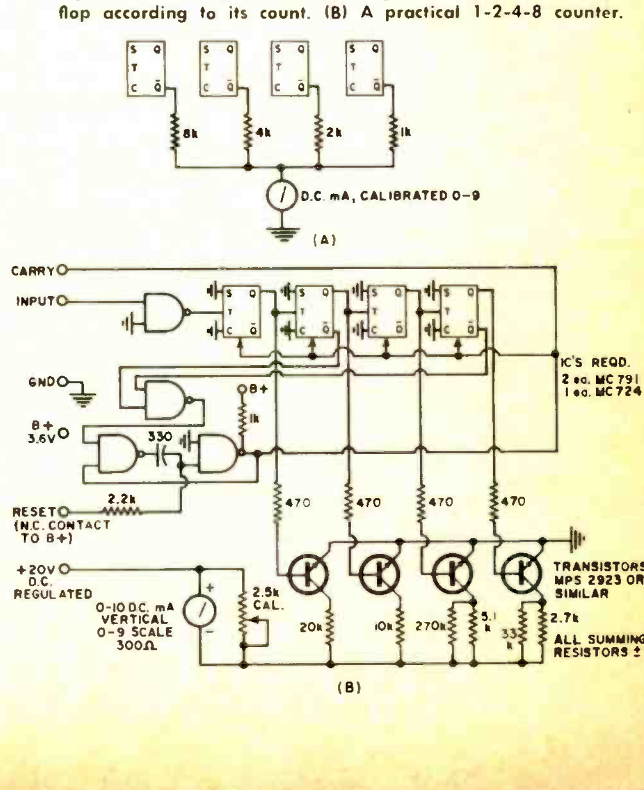
With many popular IC logic lines, the Toggle input must switch once per input pulse (in less than 100 nanosec). Thus, when counting low frequencies, the input signals must be squared to have a fall time of less than 100 ns. This is most often done with a pulse shaper. This is particularly important with RTL-type circuits.

The 1, 2, 4, 8 Counter

Fig. 2A shows how four JK flip-flops may be connected as a binary divider with sixteen possible states. If we decoded each state, we would count from 0 to 15. But if, as in Fig. 2B, we add a circuit called a *nand* gate to detect the presence of the eleventh state (count "10") and immediately preclear the counter, the binary divider could never get past the tenth state (count "9") and we would have a decade counter.

This is how that circuit operates. The tenth state is the first time, starting with count zero, that both the Q output and the second and fourth flip-flops are simultaneously grounded. The *nand* gate detects the grounding and pro-

Fig. 3. (A) Analog current summing circuit weighs each flip-flop according to its count. (B) A practical 1-2-4-8 counter.



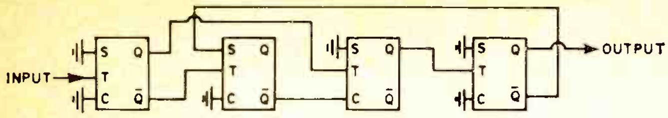


Fig. 4. Although low in cost, this modulo-10 minimum hardware circuit is difficult to decode so it is used only for scaling.

duces a positive output which preclears the entire binary divider to zero. This circuit usually works but there can be a problem. Suppose either the first or third flip-flop is a bit slow in preclearing. The *nand* gate will not detect a coincidental grounding and the counter can preset to some number other than zero. To overcome this problem, we can add a monostable multivibrator which provides a constant-width preclear pulse.

There is also a decoding problem. Ten four-input *nand* gates can be used to detect each state and produce discrete outputs for the individual counts 0, 1, 2, 3, ... 8, and 9. And by fancy circuit techniques, we can even cut down on the number of inputs many of the gates require. But this is rather expensive, especially if RTL integrated circuitry is used and additional buffers to increase drive capability are required.

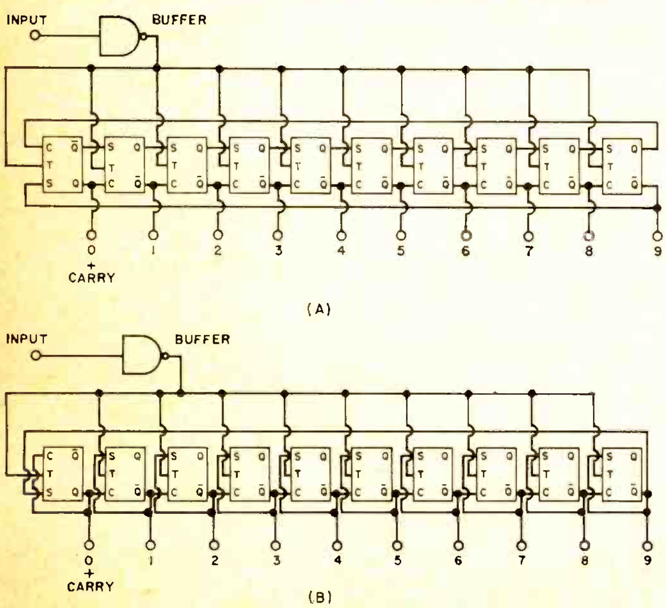
In its most inexpensive form, the 1, 2, 4, 8 counter is best used with a meter or analog readout, as suggested in Fig. 3A. Here we add 8000, 4000, 2000, and 1000 ohm resistors to the output of each JK flip-flop and measure the total current through a milliammeter calibrated from 0 through 9. Using these resistors a flip-flop counting by eight produces eight times the current of a flip-flop counting by one, the *sum* of all the currents will equal the count stored in the 1, 2, 4, 8 counter.

Problems arise in the simple circuit of Fig. 3A due to the voltage drop across the meter and temperature and supply variations. A practical form is shown in Fig. 3B which uses a higher voltage-regulated meter supply, four driver transistors, and a calibration potentiometer.

Modulo-10 Minimum Hardware

The Set and Clear inputs on a JK flip-flop are in themselves gates and we can add feedback directly to the binary divider without any extra parts and inhibit counts 11 through 16. If we do this, the modulo-10 minimum hardware circuit of Fig. 4 results. This circuit is very useful when we want to scale an input by ten, without decoding and indicating the

Fig. 5. The 10-bit shift register (A) is analogous to a 10-position stepping relay. An alternate form (B) offers easier PC layout and corrects errors. Both require six IC packages.



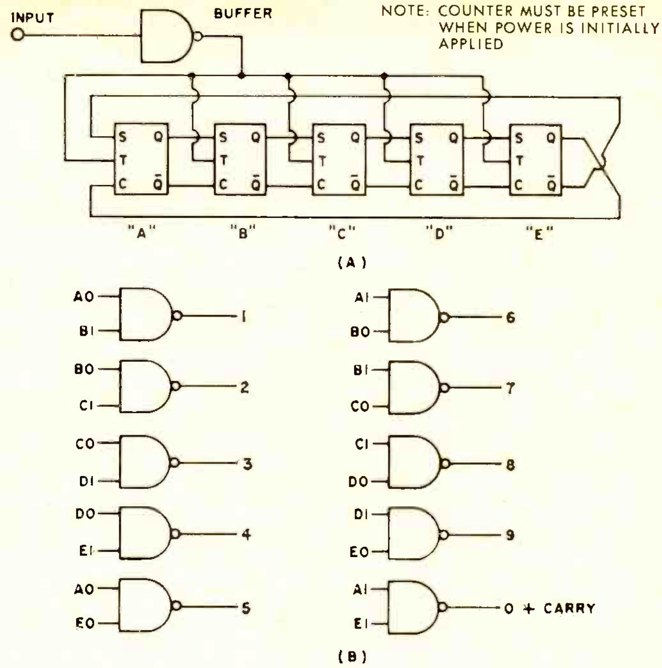


Fig. 6. (A) Walking ring counter must be preset when power is applied. (B) Ten 2-input "nand" gates may be used as inputs.

intermediate states. For instance, a modulo-10 can be used to bring a 50-MHz signal down to 5 MHz where it can easily be counted with RTL circuits. This costs far less than building a 50-MHz counter. It can also be used in counters that require precise time gates of 1, 10, 100, and 1000 milliseconds.

The 10-Bit Shift Register

By using more flip-flops, we can build a self-decoding counter. A 10-bit shift register, like that in Fig. 5A, can be used. Note that the first flip-flop is inverted which allows the counter to be preset to a 1000000000 state. The first negative toggle transition shifts the "1" one place to the right and produces a 0100000000 state. Each successive transition shifts the "1" one more place to the right, generating a sequence: 0010000000, 0001000000, and so on. We have a ten-state counter that requires no decoding. However, the output signals are at a low level so they must be amplified.

We can also change the circuit around a bit and make sure each register turns itself back to zero after it passes on the "1". Thus the register can be "self-correcting" and it lessens the chances of another "1" getting into the counter by way of a noise pulse or something similar. A buffer is normally added to the input to allow the input signals to simultaneously drive the ten parallel Toggle inputs. The details are shown in Fig. 5B.

Both these circuits have a problem not found in the 1-2-4-8 and the modulo-10 circuits. If either of these circuits ever gets into a wrong or disallowed state, it only takes few counts to get it back on the right track.

This is not true of the 10-bit shift register. When power is applied, or perhaps when a noise transient arrives, the 10-bit register can go into any one of 1024 possible states. Of these only ten are legitimate.

There are several ways to correct this. The easiest is to reset the counter every time power is applied or, if possible, immediately before each use. A second possibility is to use the presence of a "1" in the first counter to generate a brief pulse that automatically forces every other counter into the zero state. This requires another IC, a capacitor, and maybe a resistor.

We can fold a 10-bit shift register over on itself and design a decimal counter that requires only five JK flip-flops and ten two-input *nand* gates for a decoded output. The

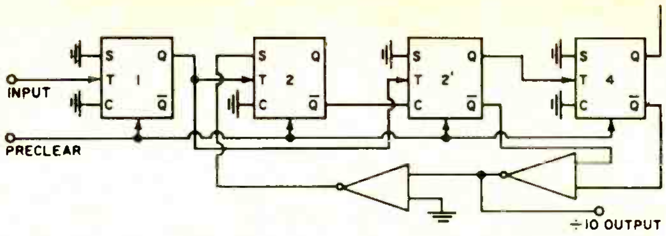


Fig. 7. In a biquinary counter, the first flip-flop divides by two. With help of two gates, other JK's divide by five.

cost will be slightly less than the 10-bit register and an output square wave at 1/10th the input frequency is available. The walking ring counter (Fig. 6) has advantages at higher speeds since only one flip-flop changes with each negative toggle transition. This is the way it works. When the counter is precleared to the 00000 state, a negative-going toggle transition will advance it to the 10000 state. The "1" comes about as a result of the crossed wires between output and input. Each succeeding input pulse generates the sequence 11000, 11100, 11110, 11111, 01111, 00111, 00011, 00001, and finally 00000. Each state is unique and decodable by a two-input gate. Transistors are required to drive the readout lamps.

Once again, there is a disallowed state problem. This particular counter can get into 32 states—of which 10 are legitimate. For example, if the counter goes into a 10101 state, on the next toggle it will go to 01010, and on the next one, right back to 10101. Thus we have a modulo-2 counter instead of a modulo-10. Worse yet, the decoding process gets mixed up and several counts at a time can appear as outputs.

But we can reset the counter to zero when power is applied or before each count is made, or both. If we must have absolutely "fail-safe" operation, we can watch for the coincidence of zeroes in the first and fifth flip-flops and use this coincidence to force flip-flops 2, 3, and 4 into the zero state. This automatically clears the counter regardless of what state it is in.

The Biquinary Counter

There is yet another way to divide by ten. Simply divide by two first, and then by five. This somewhat subtle approach is called biquinary counting and has some unique advantages. A biquinary counter needs only four JK flip-flops and with certain forms of readout, only four decoding operations are required to produce all ten counter states. This extreme economy of parts is possible only if the display can be "bent" properly to fit the counter. For a biquinary counter to be practical, both ends of the readout must be available and, usually, only visual outputs may be conveniently obtained.

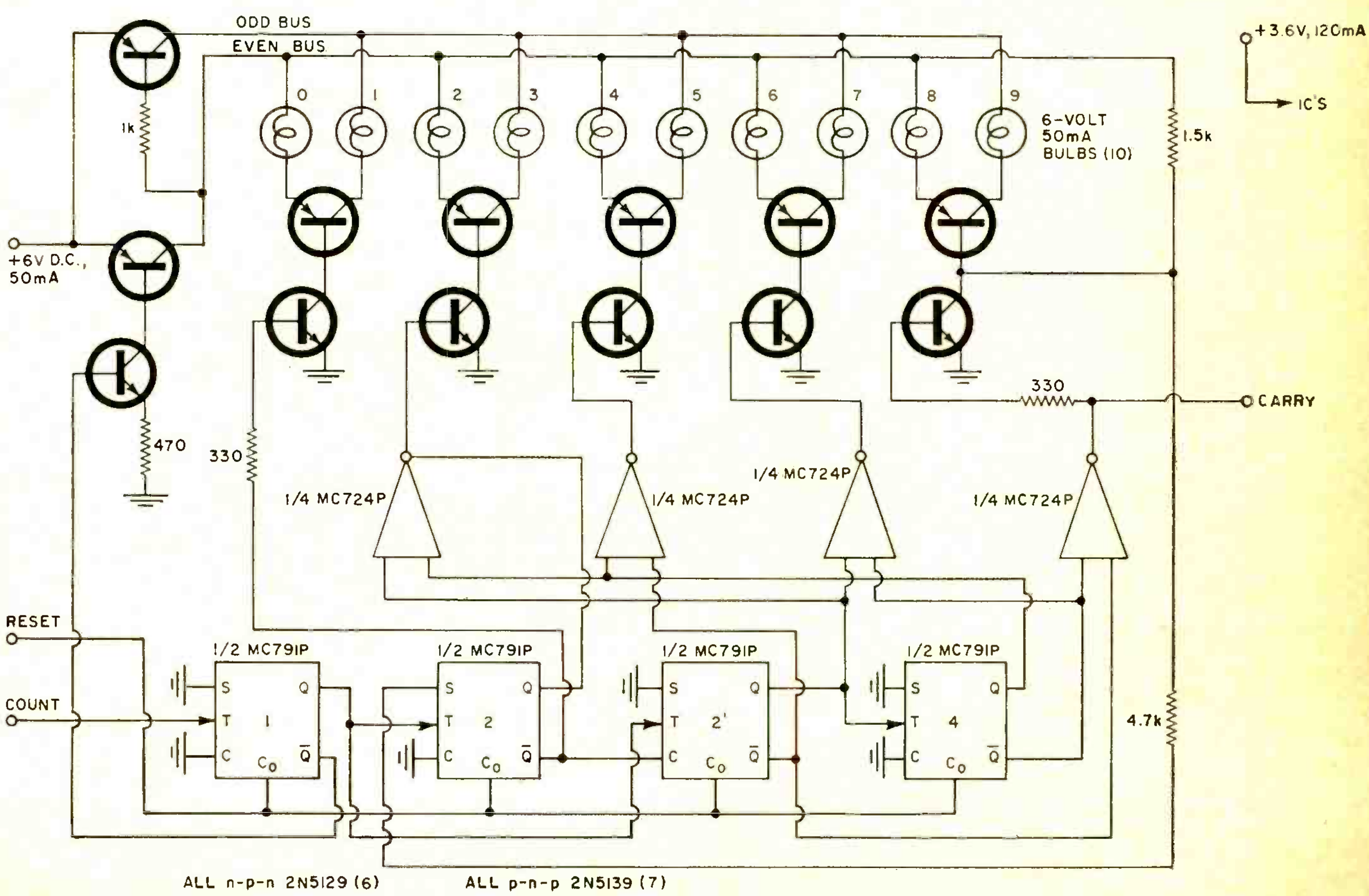
Fig. 7 shows the basic biquinary decade counter. The first flip-flop divides by 2, while the remaining three flip-flops divide by five. This particular decade counter is weighted in a 1-2-2'-4 manner.

If we added a meter readout in a circuit similar to Fig. 3, we would need two less gates, two less resistors, and one less capacitor. The resistors would be weighted 1-2-2-4 instead of 1-2-4-8 which eliminates one resistor value and makes the jump between counts 7 and 8 less drastic.

Fig. 8 shows a complete decimal counter, decoder, driver, and readout using a biquinary counter and a staggered 0-9 incandescent display. This particular circuit uses RTL integrated circuits and is useful from d.c. to beyond 8 MHz.

The outstanding features of the biquinary counter are low cost and the ease with which it can drive either an incandescent display or a special biquinary Nixie indicator. If we were to look at the counter waveforms, we would see that we have three decoded outputs: an even-odd output, a "0" or a "1" output, and an "8" (*Continued on page* 70)

Fig. 8. This biquinary counter—including decoder, driver, and readout—can be built for $10.90 per decade in kit form.



Decimal Counting

(*Continued from page 43*)

or a "9" output. Three new gates can be used to derive the remaining 2 or 3, 4 or 5, and 6 or 7 outputs.

The outputs can be combined into a readout using transistor drivers and either a group of ten lamps or a special Nixie. If we use individual light bulbs, we group the "B+" side of the bulbs into even and odd groups. The 1, 3, 5, 7, and 9 bulbs go in one group, and the 0, 2, 4, 6, and 8 bulbs go into another group. The even-odd output determines which of the two groups receive "B+" power.

The rest of the decoder outputs ground the other end of the bulbs, two at a time. For instance, on count 3, the odd bulb group is powered, and bulbs 2 and 3 are grounded. Bulb 3 lights.

The *p-n-p* transistors in series with the bulbs serve to eliminate sneak paths caused by series combinations of bulbs.

This particular counter uses only 700 milliwatts of supply power and can be built for $10.90 per decade.

Add-Subtract Decimal Counters

An add-subtract or up-down counter is one which is capable of going in either direction. Such counters are often used in calculators, computers, and positional controls. They are always more complex than unidirectional counters. There are two approaches to the add-subtract counter, the true up-down counter and the 9's complement up-down counter.

The true up-down counter behaves exactly as an add-subtract, two-coil mechanical stepping relay. In the add mode, one count is added each time. In the subtract mode, one count is removed each time. A carry output is produced every time you go from 9 to 0, and a borrow output is produced every time you go from 0 to 9. This is usually a very complex circuit, requiring either five JK flip-flops and 15 gates, or four JK flip-flops and 24 gates, not including the decoding circuit. The reasons for the complexity are twofold. In switching from add to subtract, we cannot alter the toggle inputs on the flip-flops, for if we did, it would change the count and, obviously, the count must stay the same. Second, the decoding must remain the same for both addition and subtraction. Many simple counter coding schemes do not allow this.

The 9's complement up-down counter falls into the "sneaky trick" category and gives the same results as a true up-down counter at a fraction of the cost and complexity. An ordinary decimal up-only counter is used in the add mode, and we simply apply pulses and use the carrys. To subtract, we multiply the number of input pulses by nine, and add in nine times as many events as we really have. This is the long way around, but it gives the right answer.

To borrow, the number of input counts before multiplication is compared to the number of carry pulses the counter produces. If the input pulses exceed in number the carry pulses, a borrow is needed from the next stage. Two gates, connected as a set-reset flip-flop, are normally required for this comparison. The ×9 multiplier, borrow logic, and automatic add-subtract switching take four to five IC's and add about $5 to $7 to the cost of the basic up-only counter.

The 9's complement technique's major limitation is speed. Multiplying the input pulse rate by 9 means that the speed is proportionately reduced, often by a factor of 20 or more.

Predetermining Counters

A decimal counter that can be "trained" to stop at any desired number is called a predetermining counter. Predetermining counters are used in industrial process controls, for example to count out 54,239 bottle caps. Another important application is for photographic or other precision timers where the power-line or a crystal-reference frequency is counted down to obtain a precise time duration.

Predetermining counters vary in complexity but there are four major approaches to the design of a predetermining counter. You can enter in the counter the desired number using the preclear inputs, count down to zero. When the counter gets to zero, a gate closes. Or, an up-only counter can be used and the difference between the desired number and the total possible count precleared.

The count can also begin with the up-counter at zero and a gating circuit can be used to stop the counter at the right place. While this is the most obvious approach, it is usually the most expensive and often requires very complex switches and gating.

A very simple approach is to use a walking ring counter and an additional JK flip-flop. There are ten available outputs in a walking ring, and each has one and only one negative clock transition per ten counts. Since outputs are staggered, one per count, only six flip-flops and a single-pole, 10-position selector switch are needed. The chosen negative transition toggles the extra flip-flop when the desired count comes up. But this flip-flop is enabled after the ones on the more significant decades are satisfied. For example, if we are looking for an output at 397, we enable the extra hundreds flip-flop. On count 300, this flip-flop toggles, turns around, and enables the tens flip-flop. On count 390, this flip-flop toggles, and passes on an enabling signal to the units flip-flop, which now toggles on 397, and produces the desired output signal.

This particular counter requires six flip-flops and a buffer per decade. No decoding is needed if the individual counter states do not have to be indicated.

Fully Integrated Counters

IC divide-by-ten counters and shift registers are readily available, but are still too expensive for many applications. One technique is shown by the *Fairchild* CμL958, 959, and 960 series circuits. The CμL958 is a complete 1, 2, 4, 8 counter in a single package while the CμL959 provides a momory or a strobe action that remembers the present count while the CμL958 is working on a new count. This is handy in counters and digital voltmeters where the results from the last count remain visible while the instrument is working on a new count. The final IC in the series is the decimal decoder-driver that internally converts the 1, 2, 4, 8 code into a decimal output powerful enough to directly drive a Nixie or other gas-filled readout. Other manufactures offer competing systems. But, at present, these IC's run from $8 to $25 each, making the cost per decade of a fully integrated counter with readout about $30 to $50. Complete commercial modules are priced from $60 to $100 each, with imported versions slightly cheaper. Large-quantity prices usually are far lower than this.

Today, RTL integrated circuits and discrete components are significantly lower in cost. A $10.90-per-decade cost can be realized on an RTL biquinary counter, decoder, driver, and readout. We can soon expect the prices of fully integrated counters to drop drastically and the day of a practical $5.00, one-piece decimal-counting module is in the not too distant future. ▲

Editor's Note: A complete kit of the circuit in Fig. 8, including bracket, circuit boards, bulbs, resistors, and semiconductors, is available at $10.90 per decade from Southwest Technical Products, 219 West Rhapsody, San Antonio, Texas 78216.