DESIGN and USE of a DIGITAL RALLYE COMPUTER

R. C. Rand

To those readers (probably the majority) who have never heard of a rallye, a technical description of a rallye computer might prove baffling at best and boring at worst. It therefore seems both necessary and appropriate to begin this paper with a description of a rallye and enough background to explain the necessity, or at least the desirability, for a rallye computer.

A rallye is a contest in which rallye teams, using their own cars and identical sets of written instructions, attempt to follow the course and time schedule designated by the instructions. A rallye is born when a rallye enthusiast with copious spare time and an automobile decides he would like to compose a rallye and act as sponsor or Rallyemaster. He first chooses an area, such as Howard or Carroll County, and from maps (not necessarily good ones) lays out a course of approximately 100 miles. He then drives around this course, inserting changes where desirable and choosing "clues" which he can use to write instructions from which the rallye contestants can follow his course. He next notes speed limits on all rallye roads and uses them as a guide in writing speed instructions for the contestants to follow. His remaining tasks out on the course are to choose locations for several checkpoints (at which cars will be clocked to the nearest second as they follow his course), and to measure mileage carefully between checkpoints. Finally, he writes and duplicates the instructions, cajoles some friends into acting as checkpoint crews, badgers others to pre-run his rallye to check accuracy of instructions, and arranges, through an automobile club, for a Sunday date on which to hold his rallye. It is not easy to answer the question, "Why does a Rallyemaster go to all this work?" He doesn't get paid, he uses up several weekends in preparing the rallye, he drives at least 1000 miles, and he will probably be found at the end of the rallye defending himself against from one to two hundred angry contestants. His motivation will become apparent later.

Once the rallye is announced via the rallye newspaper, the U.S. mail, or through flyers and bulletin board announcements, contestants begin registering by phone or mail. In a typical local rallye, from 10 to 50 cars will register before the day of the rallye, and another 10 to 50 will appear without pre-registering. The rallyes are scheduled for Sunday morning and start from a large supermarket or shopping center parking lot. The automobile club sponsoring the event will collect fees of $3.00 to $5.00 per car (used to provide trophies to winners) and issue numbers to the contestants which determine the order in which they will begin the rallye. The cars are started from the starting line in numerical sequence at one minute intervals, and the drivers are given a copy of the Rallyemaster's route instructions approximately one minute before departure time.

So much for the conception and administration of a rallye. To appreciate the problems faced by the contestants, it is necessary to next view the rallye as seen by the rallye team which consists of one driver and one navigator. The route instructions given the contestants at the starting line consist of from 50 to 100 instructions covering several 8-1/2 x 11-in. sheets. Instructions may vary in length from a single word "left," meaning the route turns left at the next opportunity, to an instruction that may cover half a page and require five minutes' study for correct interpretation. In addition to route instructions, there are also speed instructions so that speed at every point in the course is defined. The first dozen instructions are very easy to follow and include obvious landmarks or intersections complete with mileages given to 1/100 of a mile. These instructions constitute the "odo leg," and are furnished to allow contestants to calibrate their odometers by the "official mileages" used in computing correct "leg times." The odo leg must cover at least one-tenth the total rallye distance, and the mileages given are normally within ± 10% of statute miles and are consistent throughout the rallye. At the end of the odo leg, contestants are given about five minutes in which to compute necessary speed and mileage corrections

---

Dr. Rand received the A. B. and A. M. degrees in mathematics from Duke University and the Ph. D., also in mathematics, from the University of Maryland. He joined the staff of the Applied Physics Laboratory in 1948 as an Aerodynamicist, and later worked in Terrier dynamics, transistor applications, countermeasures, and guidance system analysis. In 1962 he became a supervisor in the Reliability and Maintainability Project of the Terrier Deficiency Correction Program, and is at present a member of the Terrier Project Office staff in the Fleet Systems Division.

Author and driver (Phil Reese) during final computer checkout.

and prepare for the timed portion of the rallye.

The division of labor between driver and navigator is left to the discretion of the team, but in most good rallye teams the driver is responsible for reading and following the instructions, while the navigator devotes his entire attention to "keeping time" and telling the driver when to speed up or slow down. The computer, which is the subject of this paper, is a device obviously designed for the navigator, but to appreciate the problems the computer must solve it is necessary to consider briefly the problems faced by the driver.

The Driver's Task

The "general instruction" of a rallye is indeed simple: stay on the road you are on until an instruction directs you onto another road, and there are no unnecessary instructions. Implicit in this is the fact that *every* instruction puts you on a new road. There are only two ways of making a driving error: the first is to leave the road you are on without an instruction, and the second is to think you have executed an instruction when you have, in fact, remained on the same road. Both mistakes are common, and the reason for the errors lies in the answer to the question asked above, namely, "Why does a Ralleymaster go to all this work?" The answer is that the Rallyemaster is a sadist who derives his pleasures from making people furious – at themselves. One example of his low cunning will suffice to illustrate the point.

Referring to Fig. 1, imagine you are at Point A on Oakland Mills Road driving at 25 mph toward B. Your next three instructions are:

(17) Turn left.

(18) Turn right at a sign reading "Route 29" and change speed to 30 mph.

(19) Turn right after a sign reading "Speed Limit 40."

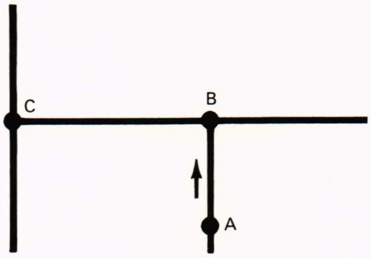

Fig. 1—Rallye route configuration.

As you approach Point B, you see a normal road sign that reads "Oakland Mills Rd." parallel to your path and "Old Montgomery Rd." perpendicular to your path. You therefore make the left turn, cross out Instruction 17, and look for a sign reading "Route 29." At Point C you see the Route 29 sign perpendicular to your path, turn right, cross out Instruction 18, increase your speed, and start looking for a 40 mph speed sign. Right? Not quite. You are going in the wrong direction on Route 29, you are travelling at the wrong speed, and you are looking for the wrong sign.

How did it happen? You made the mistake of assuming you *knew* what road you were on between Points B and C. You had every reason to believe the road was Old Montgomery Road, and you could not be penalized for believing this. But at C, had you read the other half of the road sign (that hard-to-read part parallel to your path), you would have seen that it read "Oakland Mills Rd." This meant that the left turn at B had not taken you onto a new road, and hence that you had not executed Instruction 17. Instruction 17 must then be executed at Point C, meaning you should have turned left on Route 29 and looked for the next Route 29 sign and not changed speed. This example is taken from real life in Howard County, and variations of this trap and others too numerous to mention abound in the surrounding countryside.

The lesson to be learned from this example is that your driver will not stay on course all day. If you are lucky, he will "get lost" only once or twice, and will recover quickly. In three years of rallying, often with expert drivers, I have never covered a rallye course without being off course sometime. In the bad cases, I have seen drivers cover 50% more mileage than the correct course covered.

The Navigator's Task

We are now ready to consider the navigator's task and to evaluate the necessity or usefulness of a computer. The task may be described rather simply: the rallye instructions prescribe a speed at the start, and give all speed changes in subsequent instructions. The points at which speed changes are to be executed are denoted by landmarks, rather than by mileages, and the location of checkpoints is not indicated in the instructions. The navigator's task is to compute, in real time, the car's error in either mileage or time and to be prepared to give current information to the driver as to whether the car should go faster or slower to meet the rallye schedule. The Rallyemaster computes correct "leg" times by a simple table, a portion of which is shown below:

Miles	*Speed (mph)*	*Distance (miles)*	*Time (min)*	*Total Time (min)*
32.04				0.00
	30	1.17	2.34	
33.21				2.34
	40	1.20	1.80	
34.41				4.14
	20	2.87	8.61	
37.28				12.75

The navigator would ideally like to have such a table available with mileages tabulated every 0.1 mile along the course, in which case he could easily compare time at a particular mileage with a watch, or mileage at a particular time with the car's odometer.

With the problem thus defined, various methods of computing can be evaluated. Accuracy necessary in order to win is roughly 1 second per 30 min, or 1 part in 1800. This order of accuracy is available for time (a *good* stopwatch) and mileage (using the "odo" leg for calibration). Speed can be neither measured nor maintained in traffic, so computing must be done with distance and time as inputs. Furthermore, the time or mileage comparison must be available at 10 or 15 second intervals to be useful for accurate driving corrections.

There are four methods of attack on this problem, and all rallye navigators use variations or combinations of these. The first is use of a book of tables which lists time vs mileage for various speeds; table readouts can then be accumulated to yield total time for comparison with a watch. This method obviously requires constant attention and much arithmetic and page turning. The second method is to use a slide rule to compute the correct time for convenient distance intervals. Again these answers must be accumulated arithmetically, and navigation with a conventional slide rule is difficult because of the necessity of making the mileage (and speed) correction calibrated on the odo leg. Special slide rules for rallye navigators are available in which the speed correction can be made automatically for all speeds, and a home-made version of one of these is shown in Fig. 2. The slide rule method suffers principally from lack of accuracy, and the navigator suffers eye strain from trying to read a slide rule in a moving vehicle with uncertain light.

The third method, using a Curta Calculator, is far superior to the first two methods described, and is in fact used by many leading rallye teams. The Curta Calculator is a hand-held miniature desk calculator with a hand crank. It can be set for a given speed so that one full crank yields one tenth mile (or 0.01 or 1.0 mile) increments on one readout and accumulated time on a second readout. This instrument is fast, convenient, accurate, and expensive ($100 to $200). The final, most sophisticated method, is the electronic computer or its mechanical equivalent, the "Tommy Box." This is a calculator for which the input is a mechanical or electrical signal from an accurate odometer and the output is a mechanical or electrical readout of time, the prescribed speed being a setting of three or more knobs on the computer. This type of computer requires no paper work on the part of the navigator and provides continuous accurate readouts; its sole inconvenience lies in the necessity of having the correct speed inserted at all times the car is in motion. In those cases in which the new speed is not known until the point of speed change is reached, driving time is lost while the computer speed is reset. This difficulty can be alleviated by use of two computers and a switch, but "saturation" is still possible. A refinement on this type of computer provides a difference readout which the driver can observe, thus completely freeing the navigator (to assist the driver) between speed changes.

Justification for a Rallye Computer

From the above discussion, it is clear that a computer or Curta is convenient, but is it really necessary? Wouldn't it be possible to become expert enough to win with a wristwatch and a two dollar slide rule? It wasn't possible for me, and the reason lies in the rallye scoring table. In general, timing errors are charged against the contestant at the rate of one point per second, and a maximum of either 300 or 600 may be charged at each checkpoint. In Baltimore rallyes, however, contestants are permitted to "buy" time in blocks of 5 minutes at a bargain rate of one tenth point per second. Under this system, a contestant may enter a checkpoint and request 5 minutes "bought time," and he is then charged only 30 points for this 5 minutes plus 1 point per second for deviation from the 5 minute error. This feature becomes important when, as in the example given earlier, the driver is lured off course and thus loses at least several minutes. From the navigator's point of view, the sequence of events is as follows:

The driver is proceeding merrily along requesting timing instructions from the navigator at frequent intervals. Gradually the driver appears to lose interest in timing and eventually he sadly pulls the car to the side of the road and announces: "We're lost." The navigator then bites his tongue to

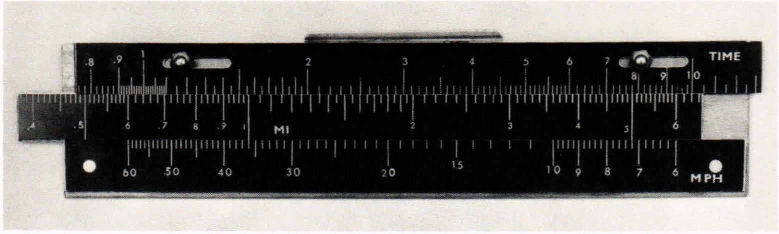

Fig. 2—Rallye slide rule.

prevent his asking, "Where did you get lost?" and stoically records the mileage at which the driver turns around. On the way back, the driver requests the navigator's help in discovering the mistake, and when they finally find the course again, the driver starts off hurrying while the navigator records the mileage and starts to compute how much time has been lost. Before he is well into the necessary arithmetic, he hears the driver saying, "Here's a checkpoint. How much time should we buy?"

This is not, incidentally, an unusually unlucky checkpoint location. The checkpoints are purposely placed (by that sadistic Rallyemaster) shortly after traps in order to prevent victims from making up lost time. Without a computer, the intelligent procedure would be to wait at the point where the mistake is found until the navigator computes lost time and then wait further until a multiple of five minutes has been lost. This procedure, while sensible, is so aggravating to the driver, who is already annoyed at being "faked out," that, practically speaking, it is impossible to detain him while the navigator does the arithmetic. A good computer must be able to compute lost time continuously while retracing, and only if this is done can the score at the next checkpoint be held to a minimum. The ability to buy time then not only dictates the necessity for a computer but also dictates a necessary design feature.

The above background hopefully explains why one would want a computer if one wanted to navigate in a rallye, but does not explain why one would build a computer. Why not, for instance, turn into a driver and let someone else worry about the stupid arithmetic? Or, if one must navigate, why not buy a Curta?

The answer to the first question is straightforward for me. I don't drive because I don't derive any pleasure from either owning cars or driving them; I just like rallyes. As to why I don't buy a Curta, I noted above that they are expensive; electronics is my hobby but spending money is not.

Since electronics is my hobby, I began to experiment with rallye computers shortly after my first rallye. I inherited and improved a poor analog computer, then built a good analog computer, then finally decided to do it right with a solid-state digital computer. I did not, however, take the approach urged upon me by my electrical engineering friends of using integrated circuits as "building blocks," since my profession is mathematics and I find computer logic to be a crashing bore. I enjoy designing circuits, experimenting with them, and building them; my rallye computer provided ample opportunity for all three fields of endeavor.

Design Philosophy

The basic computation to be made by a rallye computer is expressed by the simple equation,

$$t = d/s,$$

where d is the distance, s is the speed and t is the time. Conventional rallye computers use distance as an input, speed as a computer setting, and time as the readout for comparison with a watch. This seems reasonable since time is the variable upon which scores are based, but from my point of view a computer thus designed has two disadvantages. The first is that some electronic or mechanical input must be available from the car, and since I navigate for several different drivers, I cannot dictate the type, or even the existence of, such an input. The second disadvantage is that the computation of time requires either an electronic division (which I find difficult) or a conversion by tables from miles per hour to minutes per mile (these tables are actually used with some computers). A solution to both problems appeared to be the use of time as an input with distance as the readout, in which case the ordinary automobile odometer could be used for comparison without mechanical or electrical modification to the car. Readout accuracy is essentially identical.

Computation of correct mileage at any given time is not sufficient however for purposes of "buying time;" readout of time lost is also necessary. Since time must be available electronically as an input to compute distance, it should not be difficult to obtain a second readout of time lost, and my computer was designed with these two outputs as requirements. The finished product I visualized was a computer into which rallye speed in miles per hour is programmed, and outputs were available to display proper mileage at a given time, or, alternately, time lost at a particular mileage. Functionally, this concept is easy to conceive. It is simply necessary to have a computer output in miles which can be started and stopped by a switch, and a clock which records lost time when the computer is stopped.

This simple concept would work fine provided the driver never "made up" time and never got off course, and provided the navigator never found the computer "ahead of" the car at a speed change point. In other words, this simple concept would not work at all. We must have a "catch up" speed for the computer, and we must be able to reverse direction of the mileage readout, and to "catch up" while retracing. It is necessary, then, to have a switch with at least five positions and a clock which correctly keeps time at all five positions.

Before I had even decided how I would build this computer, I explained the concept to a rallye driver and modestly suggested it was a breakthrough in the art of rallye computer design. He quickly brought me back to earth by pointing out that I had simply designed a multiple speed, automatic Curta Calculator, in which rallye time was continuously subtracted from real time to yield lost time. Forced to grant his conclusion, it nevertheless seemed like an ideal design for my purposes.

Electronic and Mechanical Design Considerations

It is clear that the components necessary for this computer are a digital clock, an electronic circuit for multiplication, and two electrical or mechanical readouts. I considered first the circuit for multiplication.

Assuming regular clock inputs, I needed a circuit that would count accurately a specified fraction of these pulses and use the output to drive the mileage readout device. Of course the obvious choice of components was integrated circuits with which one could perform any logical computation desired. I avoided this approach because it seemed to me uninteresting, it appeared to be expensive, and I was wary of susceptibility to noise in the very noisy environment of an automobile. Alternate choices of circuits such as shift registers and binary counters used transistors, for which the above objections were generally invalid. I eventually chose a ring counter with silicon controlled rectifiers (SCR's) as basic components. I wish I could give a logically convincing argument for this choice; the fact of the matter is that on the same day that I noticed an SCR ring counter circuit I also noted a sale of SCR's at 19¢ each. Although this seems to be a poor reason for the choice of SCR's, I have since had no reason to regret it. The basic circuit that I developed is shown in Fig. 3 and operates as follows:

There are ten SCR's connected in a ring, with the circuitry of two adjacent ones as shown in Fig. 3. The SCR is a diode that conducts when the "gate" voltage exceeds that of the cathode, and stops conducting (regardless of gate voltage) only when cathode voltage exceeds plate voltage. An SCR in its conducting state is said to be "on" and a nonconducting SCR is said to be "off;" in the ring counter only one SCR is conducting, and an input pulse results in an adjacent SCR being turned on while the first is turned off. In Fig. 3, suppose SCR No. 1 is conducting. Its plate voltage is then approximately 1.2 volts, and the common cathode voltage is roughly the same. Point A is then also at 1.2 volts so that the next input pulse of 5V will result in firing of SCR No. 2, while no other SCR will fire since their input diodes have the cathode biased at 12 volts. As No. 2 SCR fires, the common cathode bus pulses to roughly 2 volts, while the plate of No. 1 SCR is pulsed downward through the action of capacitor C. Thus, No. 1 SCR is turned off a short time after No. 2 is turned on, and the circuit is ready for the next input pulse, which will fire No. 3 and turn off No. 2. This circuit has the following advantages for the purposes of a rallye computer:

1. The common cathode bus provides a steady train of low-impedance 1 volt pulses coinciding with input pulses.

2. Turn-off occurs after turn-on, so gating of cathode pulses can be conveniently controlled by the SCR turning off.

3. The circuit is simple and stable, no components are heavily stressed, voltages have a wide tolerance, and low impedance provides immunity from noise.

The gating circuit to provide passing of a selected number of pulses is straightforward. No. 8 SCR is used to turn a flip-flop off, and all other SCR's can be selected by a switch to turn the flip-flop on. While this gating flip-flop is on, diode gates pass the cathode pulses, and since No. 8 turns the gate off after No. 9 has turned on, No. 10 (or zero) always fires when the gate is off, and can be used as a trigger for a second ring counter which counts at one-tenth the rate of the first. A third ring is then added so that from a train of 1000 pulses any number from 0 to 999 can be selected by addition of gated pulses from the three ring counter cathodes.

The next problem attacked was that of readout device. Here the obvious choices were lights, "nixie" tubes, or mechanical counters. My choice of mechanical counters was based on convenience, expense, and worry about the widely varying light intensities experienced in a car during a rallye, which could make light readouts questionable. Good mechanical counters are available as surplus for approximately $2.00 each, and stepping motors designed for mechanical counters are available for approximately $16.00. Once this decision was made, the basic clock frequency could be determined.

I wished to have a computer in which a time readout would yield lost time at a mileage given by the mileage readout.

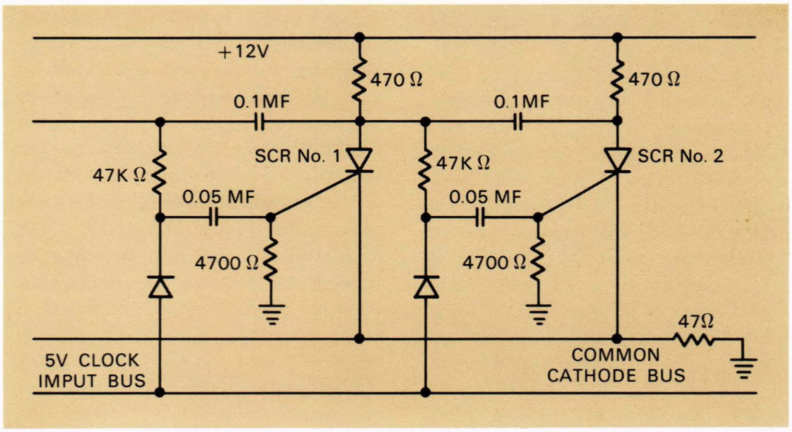

Fig. 3—Schematic diagram of ring counter circuit.

This means that at a particular rallye speed, the time taken for the mileage counter to cover a given distance must be equal to the time that would be taken at rallye speed plus the time lost. If we assume that the computer speed is m times the rallye speed s, and assume that the clock is moving at a speed of n times real time, the condition stated becomes,

$$d/ms = d/s + nd/ms,$$

which reduces to the condition,

$$n = -(m-1).$$

While m and n could conceivably vary continuously in an analog computer, it is easier in a digital computer to choose discrete values. The choice of values is based upon the necessity to accurately stop the counters while they turn at maximum rate, and $n = 4$ appeared to be as large an n as would permit this. From the above equation then, the following table of n and m is derived for control of what we shall call the "m,n switch."

Switch Position	m	n
1	–3	4
2	–1	2
3	0	1
4	1	0
5	3	–2

The interpretation of this table is simple. For $m = 1$ (Position 4), the computer is running at rallye speed, so no time is being lost. For $m = 0$ (Position 3), time is being lost at a rate equal to real time. For Position 2, the computer is retracing mileage already traversed, hence the time lost must account for both halves of the excursion, and $n = 2$. The other positions can be interpreted similarly.

With $n = 4$ as a maximum rate for the time readout, the necessary basic frequency was determined by working backwards through the counter (10 rev/min) and the motor (48 pulses/rev); 32 cycles/sec was the answer. For the mileage counter, similar computations showed 40 cycles/sec to be necessary for 300 mph at $m = 3$. An electronic multiplication of 40 by 0.8 proved an easy method to obtain both necessary frequencies from a 40 cycle/sec input. To obtain necessary inputs for the m,n switch, a "divide by 3" circuit was used for m, and two "divide by 2" circuits for n. A block diagram of the computer from ring counter input to motor output is shown in Fig. 4.

The one remaining problem was to provide an accurate 40 cycle per second input for the ring counters. In investi-

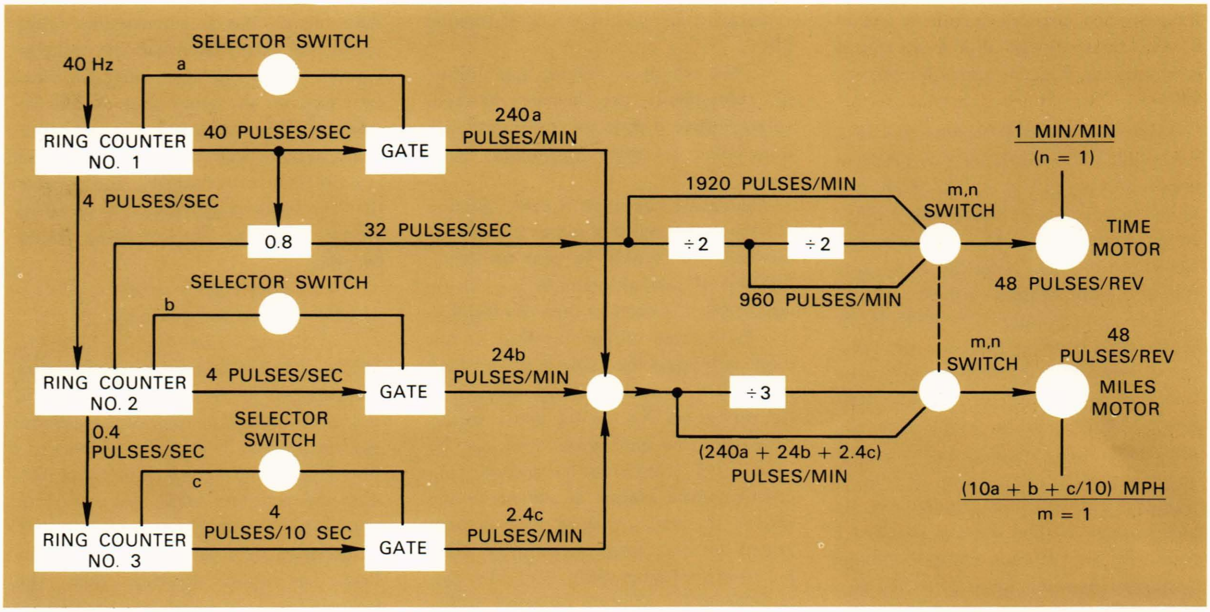

Fig. 4—Block diagram of rallye computer.

gating this problem I was impressed by the versatility and convenience of modern integrated circuits. One would naturally assume a frequency such as 40 cps could be accurately and inexpensively obtained with a suitable tuning fork, but investigation proved, to my surprise, that the cheapest way to obtain 40 cps accurately is to build a 400 kc crystal-controlled oscillator and divide by 10,000 using decade integrated circuits. It is interesting to note that this circuitry costs approximately $50.00, which is about one-third the cost of a ruggedized tuning fork, and almost exactly the price of a good stopwatch, which is not as accurate.

Physical Design and Construction

Having decided upon the block diagram and circuit details of the computer, I constructed a "breadboard" in my basement consisting of a ring counter with four SCR's, gates with three transistors, and a circuit to drive the motors using four transistors. This breadboard required little time or space and proved out the theory and circuits of the computer. I next faced the problem of final construction. I immediately noted the disadvantage (at least for home construction) of this wonderful digital circuitry; namely, it requires a large number of components and, consequently, careful packaging. A parts count revealed that my computer would require 30 SCR's for the 3 ring counters, and each SCR had 5 components associated with it. The gating flip-flops and summing circuitry would require 9 transistors, and 8 more transistors were needed for each of the two division and motor driving circuits. A financial trap was also disclosed here—my bargain 19¢ SCR's required approximately 40¢ worth of associated components for each one.

I decided to divide the physical computer into two boxes, the first one to be placed in the glove compartment of the car while the second, containing the *m,n* switch, would be held on the navigator's lap. The logical circuit division was then to place all the SCR's and gates in the first box, while the second contained flip-flops and motor drive circuitry in addition to the *m,n* switch.

I was next forced to investigate construction of printed circuit boards, since hand wiring of almost 400 components seemed to me impossible. I was surprised to find that professional-type printed boards can in fact be made in one's basement at a relatively small cost. Copper clad glass epoxy light-sensitized boards and etching solution can be purchased in small quantities at a reasonable price, and a photographic negative provides an excellent mask through which to expose the sensitized copper. The negative required is obtained by photographing a scaled-up circuit in which black tape on a white background is used to provide the desired electronic design. A four-to-one scaling makes the taped pattern easy to construct, and results in a negative too small to show slight imperfections. I made four patterns in all, from which I produced three ring counter boards 3 x 6 inches in size, two gating circuit boards 3 x 3 inches, and two dividing circuit and motor drive boards 3 x 5 inches. Packaging of these boards and associated switches into two commercially available Mini Boxes 3-1/2 x 6 x 10 inches and 3-1/2 x 6 x 8 inches was not difficult.

In designing switches and monitors I paid careful attention to the human engineering aspects of a good rallye computer. The first task was to provide some indication that the computer was in fact operating correctly, and I accomplished this by providing a pilot light that turned on and off with the last SCR in the third ring. If this light turns on once every 25 seconds, I can be sure all ring counters are operating properly, since each is triggered by the previous one. The motor circuits I could also monitor quickly, since under all positions of the *m,n* switch either the time motor or the mileage motor, or both, would be turning. The remaining circuits to be furnished were a power supply driven by the automobile battery, the 40-cycle digital clock, and a circuit to provide what the engineers call, in their curious English, "initialization" for the SCR's. Space does not permit description of these circuits; they were straightforward and uninteresting.

The final consideration was to arrange the switches so they would be convenient for the navigator without being subject to improper programming by a panicky navigator in the heat of a rallye. Two examples of hardware subject to this panic phenomenon are provided by a stopwatch and by the "Tommy Box" type computer with two speed setting dials for quick transfer. In the case of the stopwatch, it is not unusual for an experienced navigator, sometime during a four or five hour rallye, to push the wrong button and lose track of a vital time measurement. In the case of the dual Tommy Box, the navigator is likely to occasionally program the next rallye speed into the speed dials currently being used, rather than into the idle set that is to be used next.

One natural advantage of my *m,n* switch was that inadvertent switching from any one position to another would not destroy any timing information; it would merely result in producing a temporarily useless answer, such as, "if the car were one mile ahead of where it is now, it would be two minutes ahead of schedule." To exploit this convenient feature, I decided to place this switch and no others on top of the box to be held by the navigator, so that as long as he kept his hand on this switch the computed information would be available. The method of navigating is then quite simple: at the location of a speed change, the navigator simply manipulates the *m,n* switch in whatever direction necessary to set the mileage counter to the mileage of the speed change, turns the selector switch to Position 3 so that mileage stops running, changes the speed dials on the glove compartment box to the new speed, then manipulates the *m,n* switch until mileage matches the car's mileage, at which time the lost time clock will give the desired timing information. With a little practice this entire process can be performed in 10 to 20 seconds, and since rallye rules require checkpoints to be located at least one-half mile from a speed change, plenty of time is available to enable the driver to enter the checkpoint accurately.

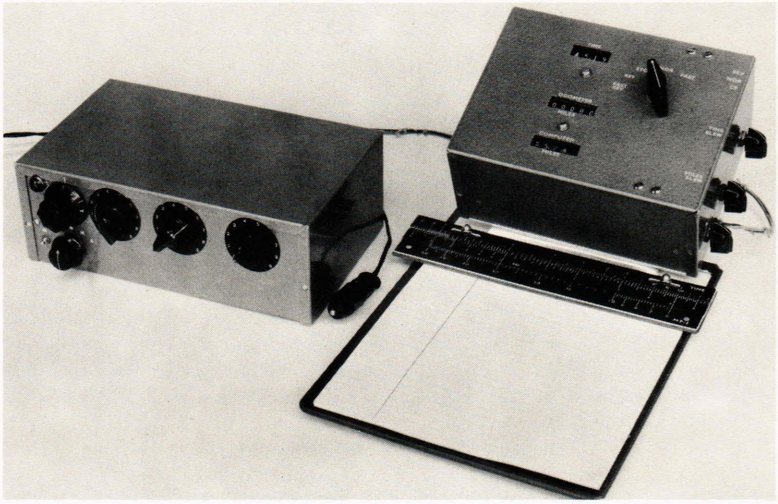

Fig. 5—Rallye computer ready for installation.

Figure 5 is a picture of the finished product. The readout box is shown on the right with the *m,n* switch and counters on top, and on the side are switches for slewing the counters and stopping the computer at a checkpoint. The "odometer miles" counter is for use when a suitable odometer pulse is available from the car. The slide rule is used for speed and mileage corrections and the tablet for keeping a rallye log.

The box on the left contains speed setting switches, and fits in the glove compartment of most cars. The switches on the extreme left are for use in ring counter initialization and monitoring circuits. Battery power enters this box from the cigarette lighter.

Conclusion

I had hoped to be able to report in this paper that I had used the computer in a rallye and found it successful, but as luck would have it I finished it one week after my last rallye of the year on November 10th (in spite of frantic efforts to finish it before the rallye). It was just as well; I used in the rallye a dual Tommy Box type of computer which, as far as I know, behaved flawlessly; yet we finished 26th in a field of 70—proving only that a good computer is not sufficient for winning a rallye. (We strayed from the course once or twice; the winners didn't.)

Except for being proven in competition, however, my computer has been adequately tested and found to be successful in every way. I have tested the design philosophy using the *m,n* switch, in rallyes with my analog computer and this concept has proved satisfactory. The digital computer has had many hours of testing in my living room as well as several hours in my car with no difficulty experienced. In a 30 minute practice rallye leg, with my 11 year old daughter running the computer and me driving, the timing error was negligible.

A question I am frequently asked, particularly by my electrical engineering friends, is whether I should not have used integrated circuits throughout, since they were so convenient for the 40-cycle clock. The answer is that if I built a second one, I would use integrated circuits to save construction time; yet I have no reason to regret not using them in the present one. There could be no improvement in reliability or accuracy and very little saving in space, since space is determined principally by the size of the switches, motors, and counters.

The final question, in which my friends have shown most interest, is whether the venture was a successful one financially. The cost in parts was approximately $100, and my time I would estimate at 400 hours. My friends gleefully point out that commercial rallye computers are available for roughly $500, so I have been slaving away at the miserly wage of $1.00 per hour. I look at it differently; even if I had no finished product, I would have had 400 hours of recreation at a cost of $100, and in these days of inflated prices, 25¢ per hour is a bargain.