Remote control of model ships

Ayukawa, K.; Foster, W. T.

Report (National Research Council of Canada. Radio and Electrical Engineering Division : ERB), 1968-12

ERB-808

UNCLASSIFIED

NATIONAL RESEARCH COUNCIL OF CANADA

RADIO AND ELECTRICAL ENGINEERING DIVISION

REMOTE CONTROL OF MODEL SHIPS

- K. AYUKAWA, W. T. FOSTER, AND F. VACHON -

OTTAWA

DECEMBER 1968 NRC # 22143

ABSTRACT

A system for radio control of model ships is described. Digital messages are sent sequentially to control the speed of three propeller motors independently with an accuracy of ±5 rpm through a range of 50 to 1000 rpm, to position a rudder with an accuracy of $\pm\frac{1}{4}$ degree through a range of 90 degrees in one degree steps, and to drive 16 latching relays which control ON/OFF functions.

Additional commands can be sent to change the working range of the propeller speeds independently on 3 motors and to change the rate of rudder motion.

CONTENTS

FIGURES

PLATES

REMOTE CONTROL OF MODEL SHIPS

– K. Ayukawa, W.T. Foster, and F. Vachon –

Problem

Model ships are tested in a manoeuvring basin measuring 200 feet by 400 feet, by 10 feet deep. The models range in length from about 16 to 20 feet, with most models about 18 to 20 feet, and are rarely under 16 feet. They weigh about 400 lb and require anywhere from 150–200 lb ballast for destroyer models to 1500–1800 lb for ore carriers. The power required ranges from $\frac{1}{2}$ to 1 horsepower and the model speed may be up to 8–10 feet per sec. The number of models having single and double screws is about equal, and a few have triple screws. Control of propeller speed with an accuracy of ±5 rpm through a working range of about 50 to 1000 rpm was considered adequate. Independent controls were required as well as the capability of reversing. The ability to position the rudder through a range of ±45 degrees with an accuracy of $\pm\frac{1}{4}$ degree was required, although an accuracy of $\pm\frac{1}{2}$ degree would still be acceptable. Another requirement was to turn other devices ON or OFF by remote control – such things as pumps, recording equipment, and so on.

The remote control gear would be located on a 35-foot tower which overlooks the basin at the mid-point of the 400-foot side. The radio link from the tower to the model would be less than 300 feet, which posed no real transmission problems but reflections from a wave-making machine spanning one end of the basin were anticipated. At first, the models would be tested under fair-weather conditions with little wind, but future plans called for year-round operation under a roof. A telemetry system would be provided at a later date.

Solution

Each command was sent as a code by the transmitter, the coded commands being sent in a sequence. At the receiving end, the start of the sequence was found and each command was separated and stored. Output actuators responded to this stored information to position the rudder, control motor speeds, and turn ON or OFF other devices.

A digital approach was taken to achieve a 'solid link'. Digital control of rudder and motor was desirable to get the required accuracy and stability. Few adjustments would be needed because the transmitter and receiver would use simple binary modulation/demodulation circuits and multiplexing would be simple. Although analog systems are inherently more straightforward, easier to service, and probably less costly, it was decided to use a digital system. One factor influencing this decision was that the experience gained would be useful in constructing a digital telemetry system.

This report gives a *general description* of the system, followed by *detailed descriptions* of each part. *Detailed diagrams* are included in the supplement [1].

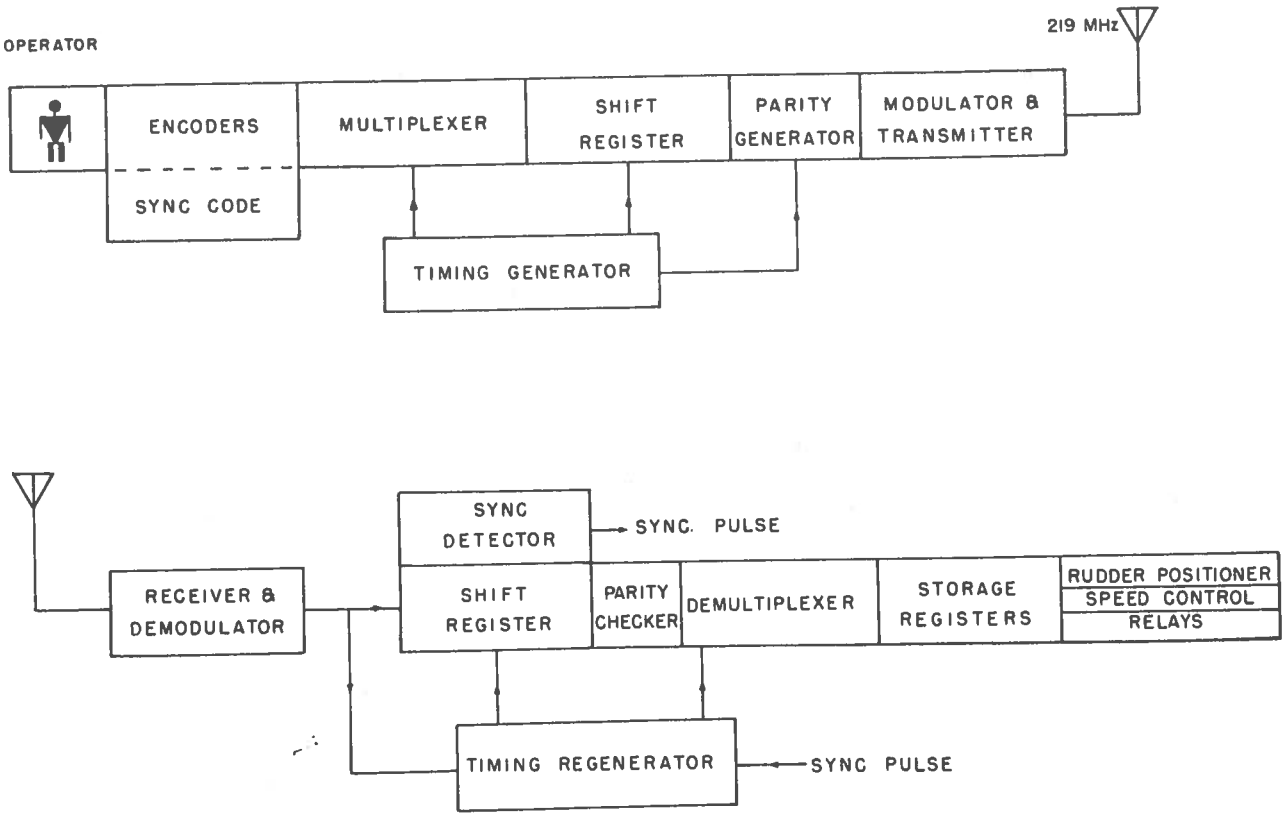


Figure 1 Remote control of model ships

Equipment – General

The block diagram (Fig. 1) shows the major parts. The transmitter will be described first.

Four encoders, one for rudder position and three for motor speed, generate the commands as each encoder is manipulated by the operator. Each command consists of 6 bits plus a sign bit, which enables the operator, for example, to position the rudder to any of the 128 positions available (120 out of the 128 are actually used) or to control the speed of 3 motors independently, each motor having 64 forward or reverse speeds. In addition, a number of push buttons (these also generate a 7-bit code) allow the operator to select ON/OFF operations, such as START/STOP port motor, START/STOP starboard motor, etc.

The sync code will be explained later. Each 7-bit code or word generated by the encoders is connected, 7 bits at a time, to the *shift register* in a sequential fashion, through the action of the multiplexer. After each word is loaded into the register, it is shifted out (serially) before the next word is loaded. These operations are controlled by a *timing generator.*

A Gray-to-pure-binary code converter and a parity-bit generator are included in the transmitter.

In order to mark the start of the sequence, a unique code word (a 16-bit frame sync pattern) is sent ahead of all other command words. This unique code is recognized at the receiving end to bring both receiving and transmitting ends into synchronism.

The receiving equipment on the model ship (lower part of Fig. 1) is a complement or a dual of the transmitting equipment just described. After the transmitter has been modulated by the serial output from the shift register, a *receiver/demodulator* recovers the original bits (with some probability of error) and puts these bits into a receiving *shift register*. Assuming that shift pulses are available from the *timing regenerator* (this aspect will be covered in detail later), a *sync detector* generates a pulse to enable the start of demultiplexing. The demultiplexer simply stores each 7-bit code word in *storage registers* and from here the various output devices, *rudder positioner*, *speed controllers*, and *relays* are controlled.

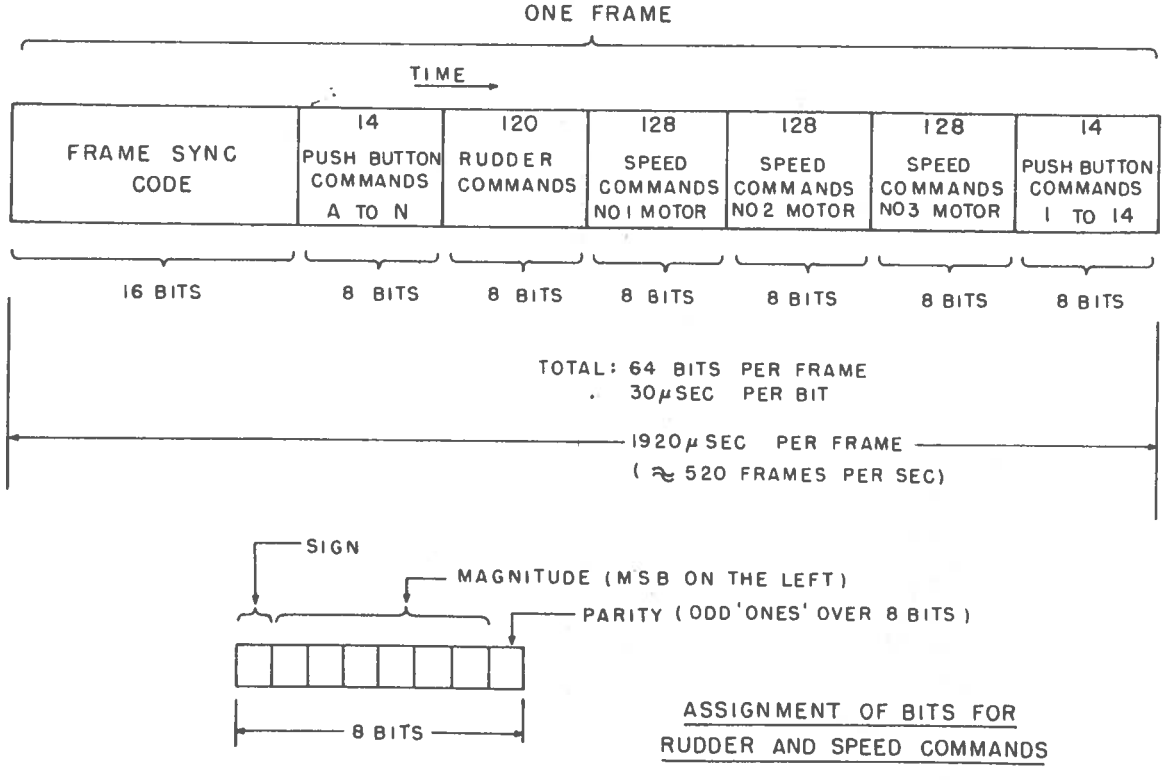


Figure 2 Frame format

It has been mentioned that the coded commands were sent in a sequence with a special code word at the beginning of the sequence. A complete sequence or frame is shown in Fig. 2. Definite positions or time-slots are allocated for each of the commands within this frame. The complete frame consists of a 16-bit frame sync code followed by six 8-bit words. This format permits the desired quantization of rudder-position and speed-control information and allows encoded information from two push buttons to be multiplexed with the other information. The addition of a parity bit for the 8th bit of each command provides some protection against single errors and simplifies the multiplexing and demultiplexing hardware.

The frame rate of 520 frames per second is quite high when one compares the number of information bits required to represent each message source (push-buttons, rudder and speed encoders) and the rate at which each message source is expected to change. This form of redundancy (albeit inefficient from the point of view of information theory), in which messages are repeated, leads to simpler hardware at the receiving end (model ship). Immunity from impulse noise was another consideration which is discussed later.

The following is a detailed description of each equipment block.

Equipment – Transmitting

Encoders (Sliding Bar)

Four identical coded bars (Fig. 3), each about 1 foot long, 1 inch wide, and $\frac{1}{8}$ inch thick, contain 7 rows of coded holes or slits. One code bar generates the rudder commands

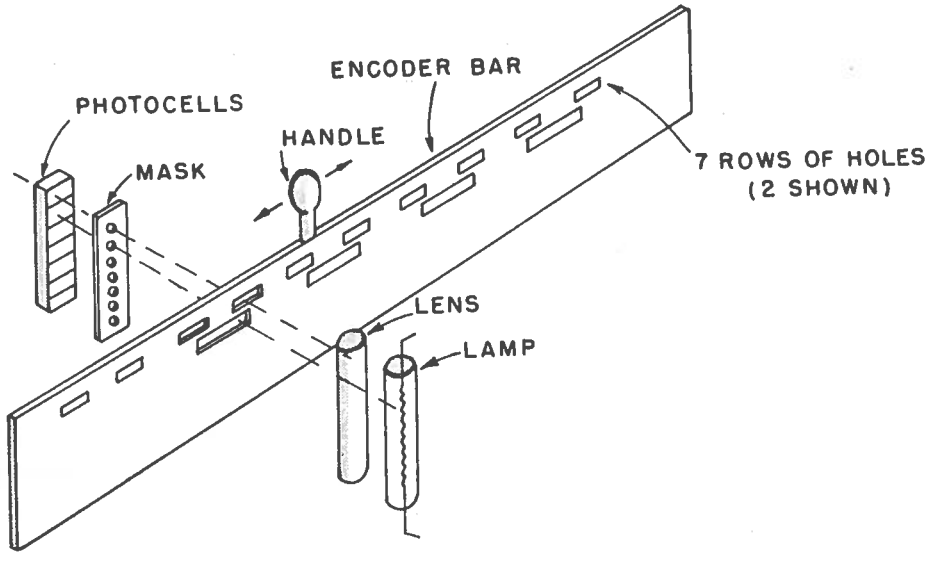


Figure 3 Sliding bar encoder

and the other three generate the speed commands. Of the 7 rows, 6 are used to represent the magnitude and one row to indicate the direction – port/starboard and forward/reverse. In each case, 64 levels or commands can be generated in addition to the direction. This allows rudder movement through a range of 0 to ±63 degrees, in one-degree steps, and motor speed control through a basic range of 0 to ±630 rpm in 10-rpm steps. This range of motor speed can be extended to 0 to ±1260 rpm if the operator so desires, by choosing 20-rpm steps. (The basic range may be set to any desired value by appropriate feedback within the speed control servo, but the number of steps remains the same.)

Conversion of the hole pattern into a 7-bit code is accomplished by a conventional light source/photo-electric sensor arrangement. The bar is coded with the Gray binary code where only one bit changes at a time when the bar is moved from one code position to the next. Thus the bar need not be centered on each code pattern by a mechanical detent, nor does each pattern need to be *exactly* aligned. The Gray binary code generated by each bar is then converted to the pure binary code. A single converter does this for all four code bars, by time-sharing or multiplexing. The converter itself is identical to the one described later under 'Rudder position control'.

To summarize, the operator has within easy reach, four joysticks which permit him to encode whatever commands he wishes by simply moving the encoder bars. Each bar has attached to it a pointer which slides against a scale graduated in decimals, showing rudder angle in degrees or tens of revolutions per minute.

Encoders (Push-button)

The object of the push-buttons is to allow the operator to make a number of ON/OFF type selections on the model ship. An example already mentioned was to start or stop the port or starboard motors. Other ON/OFF functions include such things as STOP ALL MOTORS, X1 or X2 (rpm scale changed by remote command to give ±630 rpm, full scale, or ±1260 rpm, full scale), SLEW RATE OF RUDDER – SLOW or FAST, etc.

Each push-button, when depressed, produces a 7-bit parallel word. Fourteen push-buttons are labeled by letters A to N, and another set are labeled by numbers 1 to 14. To initiate any remote action, 2 buttons are pressed (order does not matter), for example, A1 yields ± 630 rpm full scale on motor A, A2 yields ±1260 rpm full scale on motor A; and, similarly, B1, B2 and C1, C2 allow the full-scale speed to be changed at will on motors B and C. The total number of selections, such as A1, A2, A3. . .A14 through N1, N2, N3. . . N14 is 14 X 14 = 196. Thus, a total of 98 ON/OFF functions can be performed.

Each of the 14 code words associated with the letters A to N, should be as different from each other as possible. Each code word consists of 7 bits plus a parity bit to make the number of ONES odd. Thus with 8 bits, there are 256 possible code words and out of these, 14 code words are selected that are mismatched as much as possible. In this way, an errant noise pulse (or pulses) would have less chance of changing one command into another.

The construction of this set of code words is described next. The basis is a Hadamard matrix.

First a group of 4 bits are arranged as shown in Fig. 4(*a*).

1 0	1 0 · 01	1001 · 0110	0110 1001
1 1	1 1 · 00	1100 · 0011	0011 1100
(*a*)	· · · · · ·	1010 · 0101	0101 1010
	1 0 · 10	1111 · 0000	0000 1111
	1 1 · 11	· · · · · ·	
	(*b*)	1001 · 1001	0110 0110
		1100 · 1100	0011 0011
		1010 · 1010	0101 0101
		1111 · 1111	0000 0000
		(*c*)	(*d*)

Figure 4

In the second step, Fig. 4(*b*), the 4 bits are written into the 4 quadrants. The 1st quadrant contains the complement of the initial 4 bits. The 2nd, 3rd and 4th quadrants contain the initial 4 bits, unaltered. This procedure is repeated to produce the 8 words shown in Fig. 4(*c*). It will be noticed that the first column on the left contains all ONES which could be construed as the parity bit (even) for each of the 8 code words. If odd parity is desired (i.e., an odd number of ONES) this column could be all ZEROS, without affecting the number of mismatched bits between any pair of code words.

Another set can be constructed by simply complementing the set shown in Fig. 4(*c*). This set, Fig. 4(*d*), has the same properties as the other. Evidently, *at least* 4 bits are mismatched between all pairs of words. By deleting the all ONES and all ZEROS (the last rows) word, 14 different words are constructed. As mentioned earlier, the first columns can be either all ZEROS or all ONES to make odd parity.

Each of the 14 code words is also associated with one of the numbers on the 2nd set of push-buttons. Thus, when the selection A1, for example, is made, remote action is initiated only when all 14 bits (plus the two parity bits) are received correctly. This has the desired effect of reducing the probability of false alarms.

Multiplexer

The function of the multiplexer is to take the information from the input devices (bar and push-button encoders, toggle switches) and load them into the shift register. This is done sequentially through the action of pulses $P_1, P_2 \ldots P_8$ which come

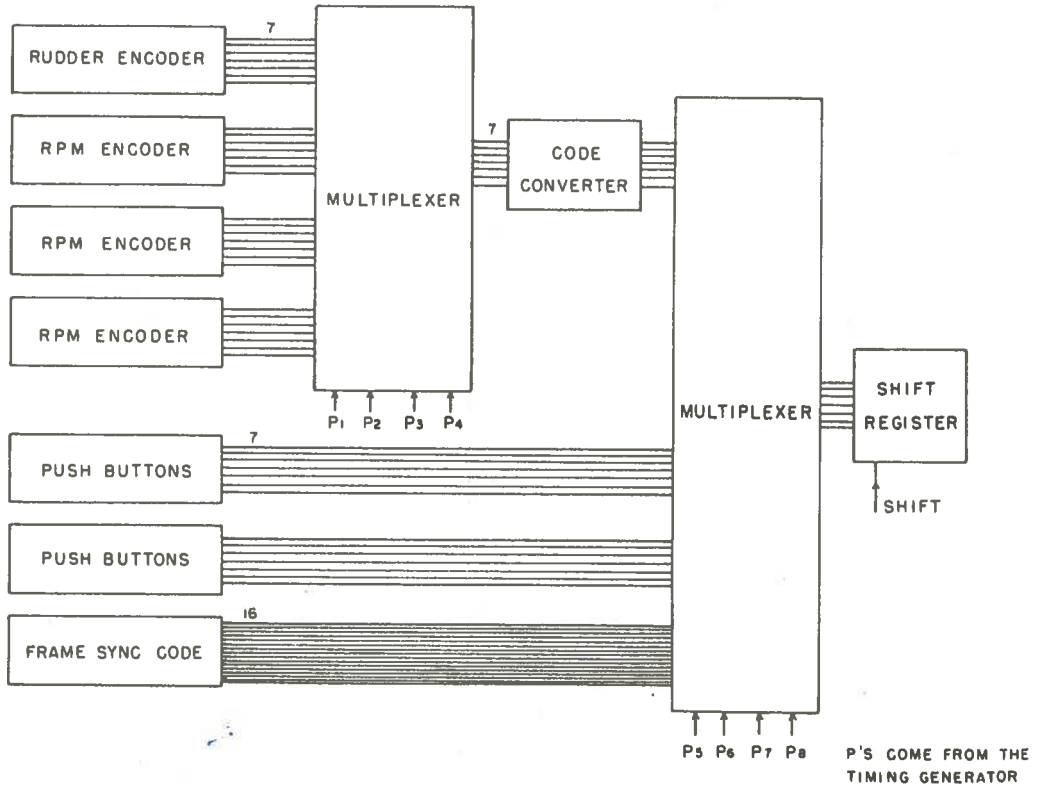


Figure 5 Multiplexing the input devices into a shift register with pulses P_1 to P_8

from a timing generator as shown in Fig. 5. The order in which the input devices are multiplexed is shown in Fig. 2 (frame format). The complete sequence is continually repeated as the contents of the register are shifted out serially after each parallel load.

A logical diagram, Fig. 6, shows the components required for the upper left multiplexer. This is simply a series of AND/OR gates which connect the 7 bits from each encoder (Gray coded) to the code converter. The connections are conditional upon P_1, P_2, P_3, and P_4. The lower right multiplexer is essentially the same as the other.

The frame sync code (16 bits) is loaded into the register, 8 bits at a time. The code itself is generated by toggle switches. Even parity is used.

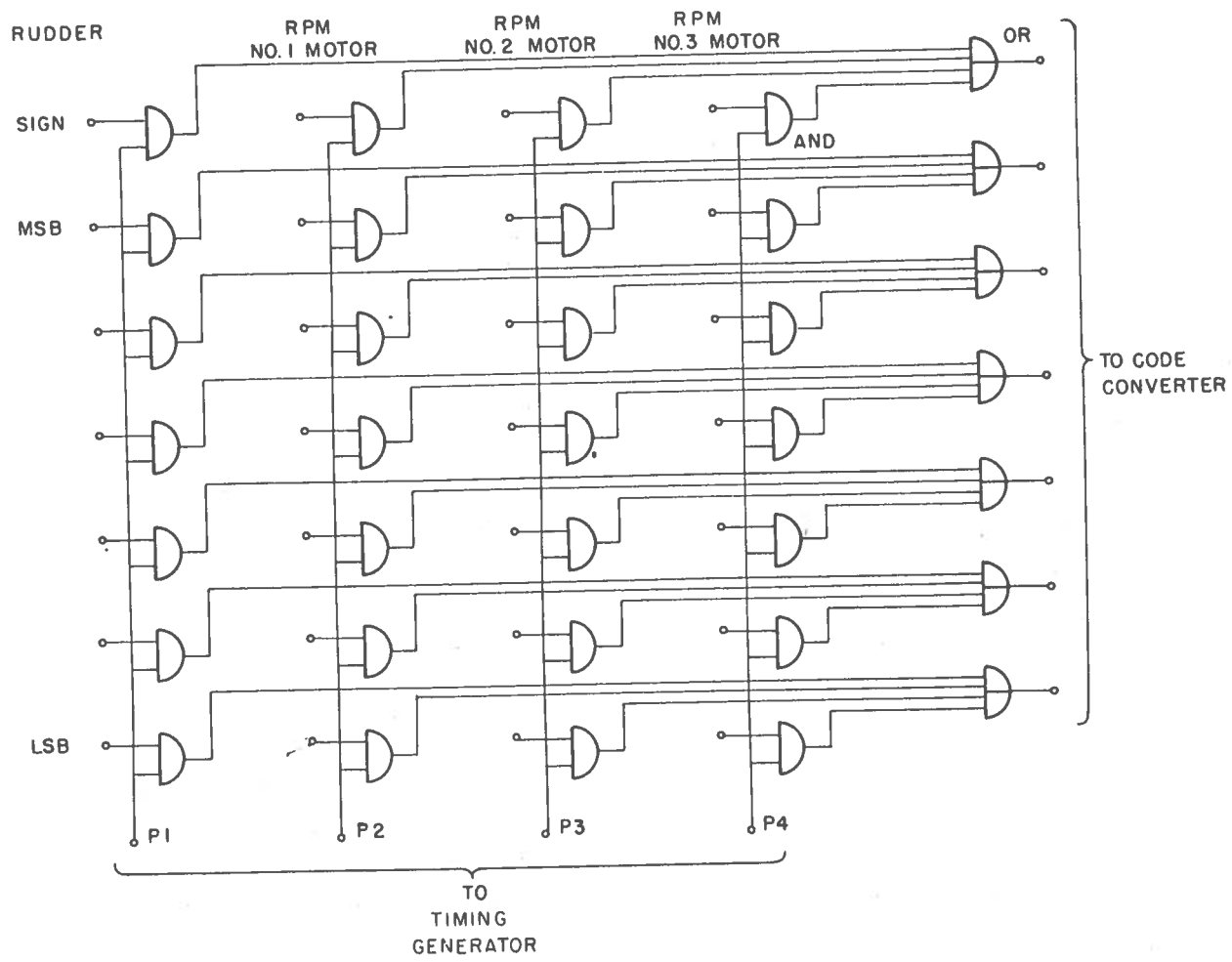


Figure 6 Logic diagram of multiplexer for rudder and rpm encoders

Shift Register

In this application, the function of the shift register is to convert the parallel information coming from the multiplexer into serial form. Also at this point, it is convenient to generate a parity bit for each of the coded commands and insert this bit into the serial stream.

The timing of the events is as follows. A 7-bit word (command) is loaded into the register, parity is then generated to make odd ONES and the resulting 8-bit word is shifted out. When the last bit has been shifted out, the next word is loaded, and the whole process is repeated.

The 16-bit frame sync code is made up of two identical 8-bit words. The 8-bit word was chosen to have even ONES or even parity and so, in this case, when the frame sync code is loaded into the register, the parity generator is disconnected. The reason for having even parity for the sync code and odd parity for the command words is to simplify the operation of the sync detector in the receiving end equipment.

It is desirable to have odd parity in the message portions of each frame so that long strings of ONES or ZEROS do not occur. This helps in the recovery of bit-timing at the receiving end.

The shift register output now consists of a 16-bit frame sync code (even parity), followed by six 8-bit (odd parity) messages or commands, giving 64 bits per frame. These bits emanate from the register in a steady stream with no spaces in between the bits (non-return to zero).

No further coding is applied to the individual bits or to the messages. This 'raw' bit stream then enters the modulator, which simply turns ON a CW carrier for a ONE and turns OFF the carrier for a ZERO. (The details of the transmitter and receiver circuitry with a description is given in the section titled Transmitter and Receiver.)

Timing Generator

As mentioned earlier, each 7-bit code or word is loaded in parallel into the shift register. After each word is loaded into the register, a parity bit is generated and this is inserted as the 8th bit. This 8-bit word is shifted out completely before the next word is loaded. These operations are controlled by a *timing generator* (Fig. 7).

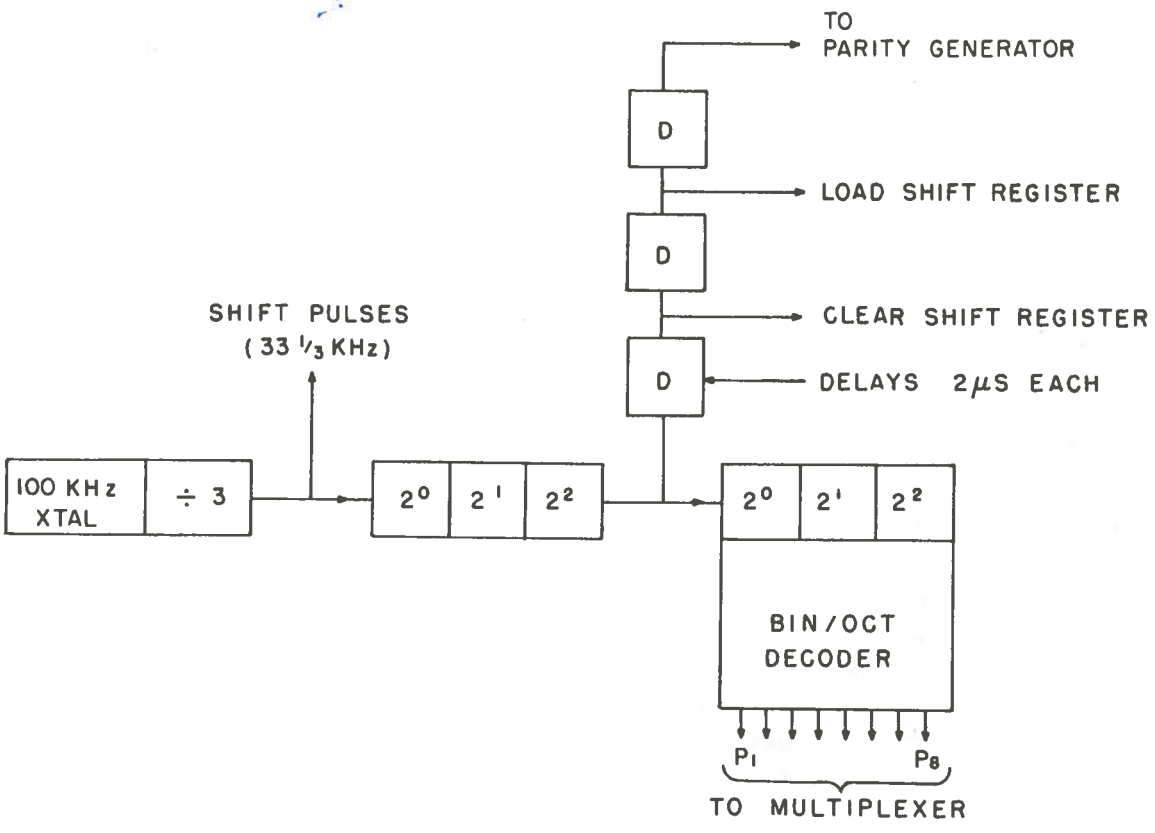


Figure 7 Timing generator

A few other details could be mentioned. These are listed below.

Parity — This is computed only for the command words to form odd parity. For the frame sync code, the first 8 bits are identical to the second 8 bits, and both parts have even parity. The parity generator is therefore by-passed twice per frame for the sync code.

Clear — The shift register is cleared prior to each loading. This is evident in Fig. 7. CLEAR occurs every 8 shifts.

Shift — The serial shift pulses occur every 30 μsec and shift the register continuously. The bit-time is thus 30 μsec long.

Delays — The delays insure that the clearing, loading, parity generating, and shifting do not occur at the same instant.

Clock — A 100-kHz crystal-controlled oscillator is divided down to $33\frac{1}{3}$ kHz to form shift pulses spaced 30 μsec apart. This clock can be slowed down for manual checking of the operation, or it can be stopped and each step in the operation advanced by a push-button.

Displays — These enable the operator to observe the commands or words, plus the parity bit, etc., before being shifted out.

To summarize, the frame format is simply as follows: a 16-bit frame sync code followed by six 8-bit words, for a total of 64 bits per frame. Each bit is 30 μsec long in the non-return to zero (NRZ) mode producing about 520 frames per sec.

Modulator/Demodulator

The model basin (200 feet X 400 feet) has a 35-foot tower from which the remote-controlling would be done. This established the transmitter and antenna site. The remaining problems were (1) estimating the transmitter power required for covering the basin with sufficient signal even in the nulls, and (2) choosing a modulation scheme compatible with the transmitter and the modulating waveform.

The first problem, that of estimating the transmitter power, was resolved by direct experimentation. The received signal strength was observed over the entire basin, particularly in the area close to the wave-making machine. The machine spans one 200-foot end of the basin and has metal plates partially immersed in water and the rest (3 to 4 feet) above the water line. With a transmitter power of 50 mW, and a receiver with a sensitivity of 5 μV, it was found that the output signal was nicely limited, or 'saturated'.

Even when the transmitter power was reduced by 20 decibels, there was satisfactory output signal over the usable area of the basin. With this reduced transmitter power, the nulls became noticeable – the output signal would take a sharp dip as the model ship passed through the null. The sharpness and depth of the null were difficult to measure; attempts at positioning the receiver in the nulls were frustrated by small movements of the model ship (rolling) causing large changes in the output signal. In any event, it was concluded that even in the worst case (model ship close to the wave-maker and receiving antenna in a null), a transmitter power of 50 mW would certainly be enough, and that over the working area of the basin, this power would provide a margin of at least 20 db over the minimum requirement.

In conjunction with this, two elementary methods of modulation were tried. The first method was frequency-shift-keying (FSK) where two separate carriers at slightly different frequencies were alternately turned ON and OFF by the ONE'S and ZEROS. The advantage of FSK – using a second carrier at a slightly different frequency to fill in the nulls which may occur with one carrier – did not materialize. This method was dropped in favour of the simpler AM method where a single CW carrier was turned ON for a ONE and turned OFF for a ZERO. The circuitry for this modulation/demodulation became simpler, and, as mentioned before, there was a good margin (20 db) of signal strength with a 50 mW transmitter.

The choice of carrier frequency (219 MHz) was based upon the small antenna size and the relative ease of generating and modulating a carrier at this frequency. Crystals at both ends, transmitting and receiving, provided the stability against drift.

Noise, mainly very short (2–3 μsec) impulses, came from a gasoline powered engine generator which provided the basic power (110 V, 60 cycles) for the remote control equipment and the propulsion and rudder motors. These short, high-energy impulses were allowed to go through the wide-band receiver without 'stretching' as the binary signal (carrier on/off) would be contaminated for just a short time, the length of the impulse being much shorter than the bit-time (30 μsec). This provided some protection against impulse noise.

SUMMARY OF MODULATION/DEMODULATION

At the transmitting end, a 220-MHz carrier is turned on to represent a binary ONE and turned off for a ZERO. The ON or OFF interval is 30 μsec.

At the receiving end, the presence or absence of this carrier is detected at the output of the IF amplifier by a simple envelope detector. This is followed by a video amplifier whose output is 'sliced' to recover the original binary waveform.

Equipment – Receiving

Basic Receiving End Operations

The receiving end equipment must 'obey' the commands which have been transmitted to it. These commands are sent much faster than they can be obeyed or responded to. This means that while the rudder, for example, is moving towards the commanded position, the information regarding this position must be available on the model ship; i.e., the rudder commands must be written down or stored on the ship. Also, as these commands are changed or updated, the old commands must be erased, or cleared, before the new commands are stored in their place. Similarly, speed commands are stored in three storage registers, one for each of the three speed commands. Push-button encoded commands are also stored, then decoded, to actuate latching relays (See lower part of Fig. 1).

Thus, all the transmitted commands are stored in six separate registers; 1 register holds the digital representation of the desired rudder angle, 3 registers hold the speed information, and the last 2 registers hold the push-button encoded commands. It is from these registers that the output actuators (rudder controller, speed controller, relays) get their command information.

Stated briefly, these commands, which are transmitted in a definite sequence or order, must be sorted out at the receiving end and then deposited in their respective storage registers. The first step is to recover timing, and then the sorting and storing of the commands follows immediately.

Recovery of Bit, Word, and Frame Timing

BIT-TIMING

Shift pulses are required to shift the incoming bits into a shift register. These pulses must occur at the same rate ($33\frac{1}{3}$ kHz) as the incoming bits and be phased in such a way that a shift occurs somewhere near the middle of the bit (or near the end of the bit, but not at a transition of a bit).

Referring to Fig. 8, the second waveform shows the incoming bits in the non-return to zero (NRZ) mode. From the transitions in this waveform, short (2 μsec) pulses are produced. These are impressed on a high-Q LC circuit (tuned to $33\frac{1}{3}$ kHz) which starts to ring as shown. The peaks of the sine-wave are coincident with the driving pulses. If this sine-wave is squared up, the required bit-timing waveform is produced as shown on the bottom line. Note that this waveform is displaced from the original bit-timing waveform.

The positive transitions of this regenerated bit-timing waveform occur at an instant of time which is $\frac{3}{4}$ of 30 μsec away from the beginning of any information bit. If these transitions are used to shift the incoming bits into the shift register, the bits are effectively sampled at those instants.

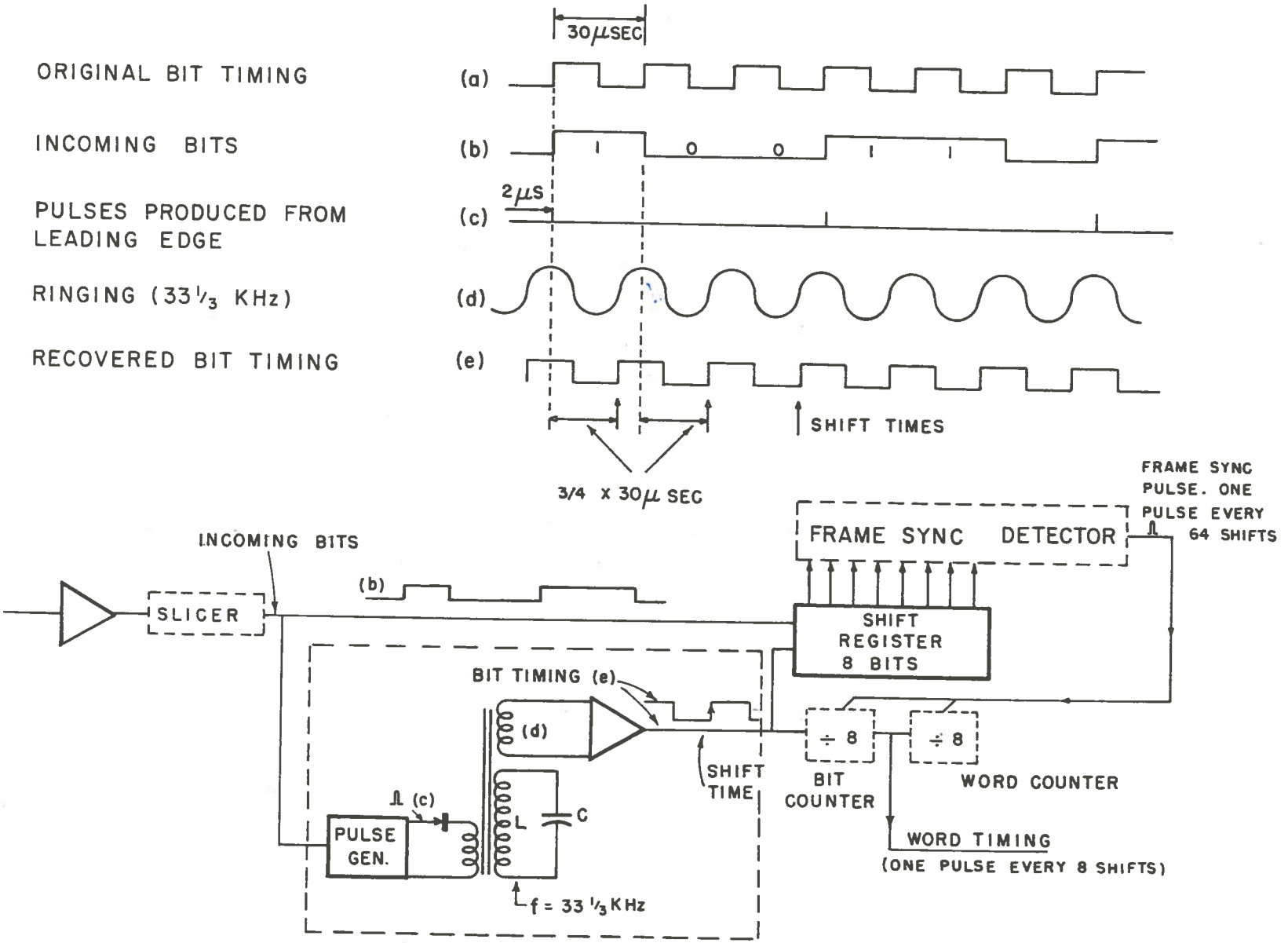


Figure 8 Recovery of bit timing

WORD-TIMING

The 'word-time' is the time at which the shift register is filled with a complete 8-bit word. This occurs after every 8 shifts and a read-out pulse is generated at these times to transfer the contents of the shift register into 6 storage registers.

For example, in Fig. 9*b*, when the 8 bits of word No. 1 are in the shift register, a read-out pulse is generated immediately after the 24th shift. This pulse occurs at shift-time t_{24} and initiates a parallel transfer of word No. 1 into storage register 1. Similarly, at t_{32}, i.e., 8 shifts later, word No. 2 is transferred into storage register 2.

Thus, in order to make the 6 transfers (there are 6 words) possible, word-timing pulses are required at shift-times t_{24}, t_{32}, t_{40}, t_{48}, t_{56}, and t_{64}, i.e., after every 8 shifts. These pulses are produced by modulo 8 counting of the shift pulses.

All transfers are conditional upon a parity check which takes place at times t_{24}, t_{32}, . . as noted above.

FRAME-TIMING

A synchronizing pulse is required to synchronize the demultiplexing operations at the receiving end to the multiplexing operations at the transmitting end. Specifically, this pulse clears the bit-counter and the word counter – thus insuring that the *start* of the count will occur at the correct instant. This in turn means that the 8-bit words would be completely inside the shift register before being transferred, and that these words would be deposited in the correct storage registers.

A frame consists of a 16-bit frame sync pattern (2 identical 8-bit sequences) followed by six 8-bit command words, giving a total of 64 bits per frame. The object of the frame sync detector is to recognize that the two 8-bit sequences have entered the shift register, and when this occurs, to produce a single pulse (frame sync pulse) indicating the beginning of the frame.

This is done in two steps (Fig. 9*a*). In the first step, the shift register contents are compared (in parallel) against a stored replica of the expected 8-bit sequence; if the comparison shows good agreement, then a pulse is produced. This first pulse is retained or stored (by means of a delay) to be ANDED with a second pulse. In the second step, another pulse is produced in the same manner as the first, only in this case, the second half of the frame sync pattern is compared against the same stored replica. If this second pulse arrives in coincidence with the first pulse, this is announced by a final pulse, the frame sync pulse.

The comparisons are made after each serial shift. Eight bits are compared at a time. Like bits put a positive voltage on the ends of the resistors, whereas unlike bits put a negative voltage on them. The output voltage is the sum of the positive and negative currents flowing through the output resistor. If all 8 bits agree, i.e., each bit in the register agrees with each bit in the stored replica, 8 units of voltage appear at the output. If 7 out of 8 bits agree, there is a net voltage of 7 minus 1 or 6 units. For 4 agreements and 4 disagreements, there is a net of zero. Consequently, the threshold (zener diode) can be set for different degrees of agreement (or disagreement). For example, if we allow one disagreement, then an output will be produced when the frame sync pattern has single errors. For this case, two or more errors will produce no output.

Figure 9*b* shows the time at which these pulses occur. The first pulse occurs at t_8 and the second pulse occurs between t_{16} and t_{17}. The final pulse occurs at t_{16} (at the same relative position in every frame) to clear the bit counter and the word counter at the start of each frame. If for any reason the frame sync pulse is missing (due to noise), the two sets of counters will continue to count in modulo 64 and thus retain synchronism with the multiplexing at the transmitting end.

Demultiplexing and Storing Commands

Figure 10 shows the details of the demultiplexing process. First, a 3-bit decoder attached to the word counter provides the sequential gating waveforms into the inputs of six 2-input AND gates. (Two of the 8 outputs of the decoder are not used since the first two words are associated with the frame sync pattern). The other inputs of the AND gates are fed with read-out pulses; these come from the output of the lower left AND gate whose inputs are the delayed word-timing pulses and the parity checker. The delay (2 μsec) is necessary to prevent a read-out while the shift register is shifting or while the word counter is counting.

The parity checker evaluates the oddness or evenness of the number of ONES in the shift register. This is done after every shift. However, it is only at each word-time, i.e., the instant at which the shift register is filled with a complete word, that the output of the parity checker is used. Hence, parity is evaluated after each 8-bit word has entered the shift register completely. If parity is shown, i.e., an odd number of ONES in 8 bits, then a word-timing pulse is allowed through the lower left AND gate.

The six AND gates provide read-out pulses which effectively load the storage registers with the contents of the shift register. The loading pulses are shown, spaced 8 shifts apart and slightly delayed from the shift times t_{24}, t_{32}, etc.

Thus the main object of sorting out the commands which are being sent sequentially, and depositing the commands into storage registers, has been achieved. This process has

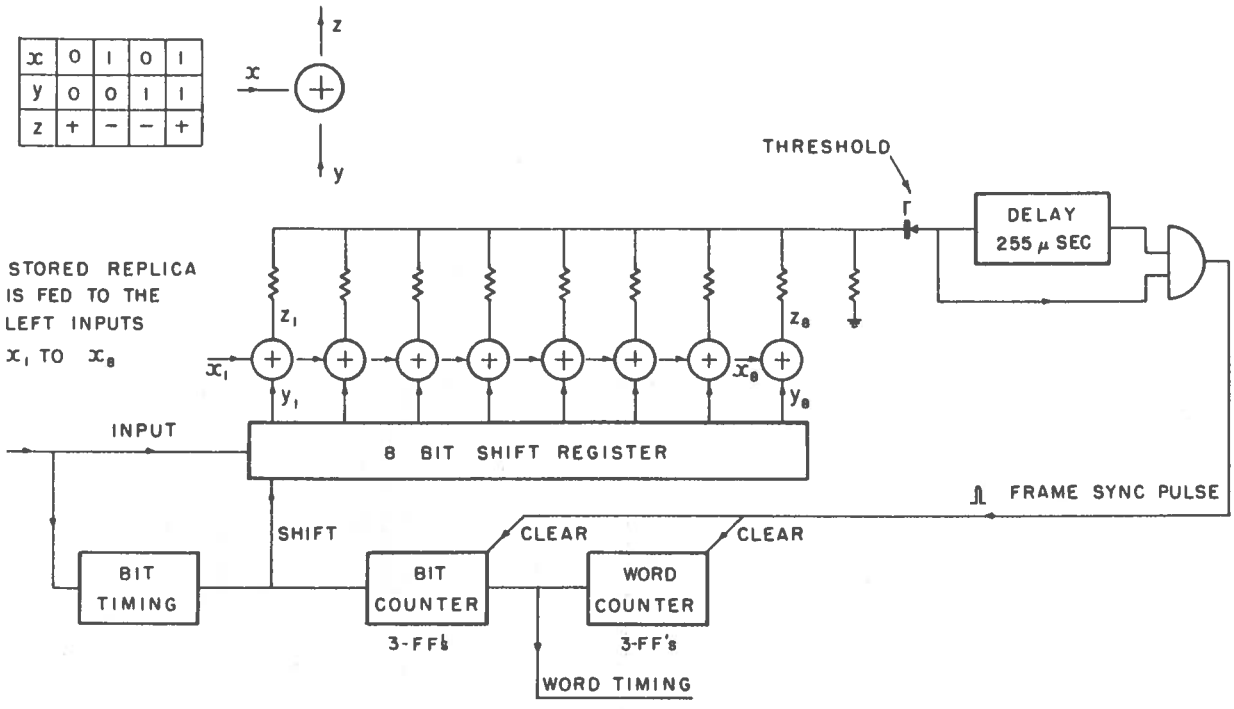


Figure 9(a) Recovery of frame synchronizing pulse

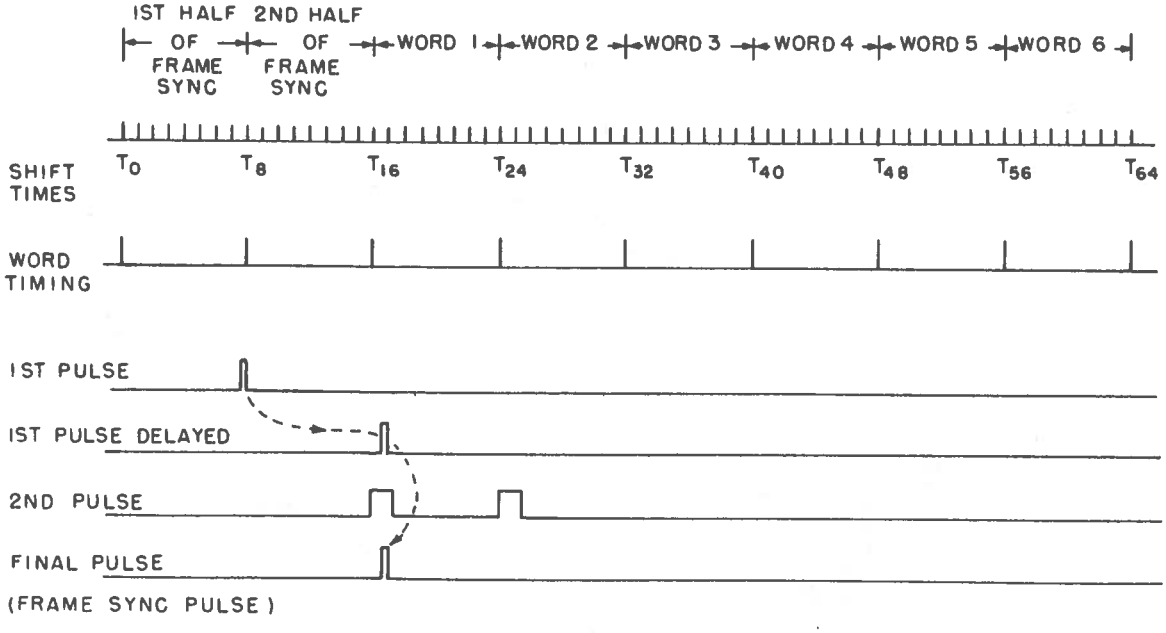


Figure 9(b) Timing waveforms for word and frame synchronizing

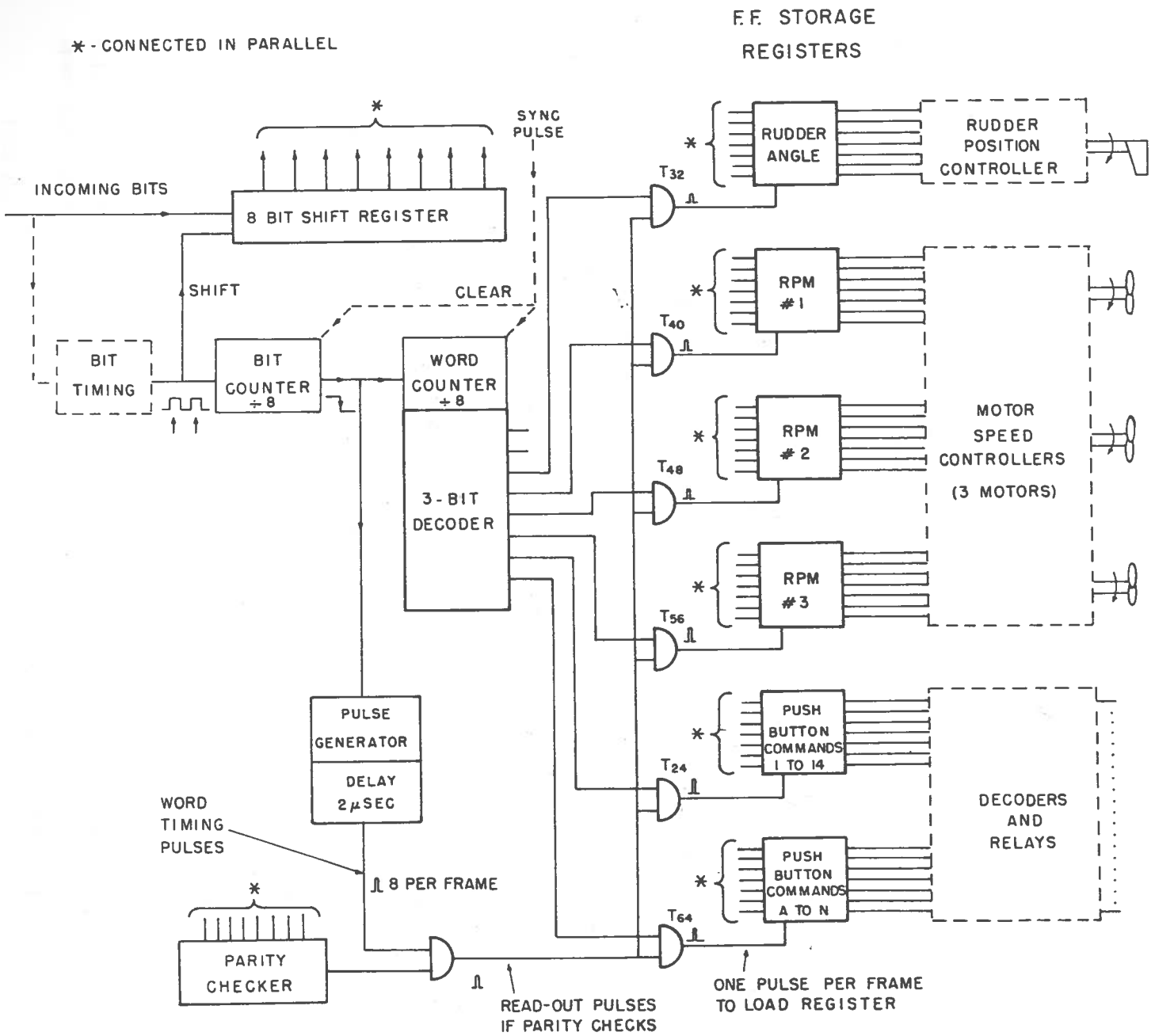


Figure 10 Demultiplexer

required the extraction or recovery of timing information – bit, word, and frame timing – which had been embedded in the transmitted waveform. The recovery of this information has depended on prior knowledge of such things as the bit rate, the frame sync pattern, the number of bits per word, and the frame format.

Rudder Position Control

The requirement here was to position the rudder through an angle of ±45 degrees in discrete steps. For small rudder angles in the range 0 to ±10 degrees, it was desired to have steps of one degree, and for angles larger than 10 degrees, to have a step size of 5 degrees. The torque load was given as one foot-pound at a rudder slewing rate of 90 degrees per second. The final resting position of the rudder was to be held within $\pm\frac{1}{4}$ degree or better. Control of the slewing rate was considered to be desirable but not essential.

The basic idea in the solution of this problem was to use a stepping motor to move the rudder, and a shaft angle encoder to provide the feedback information to a comparator. Such a servo was built and it is shown in the block diagram, Fig. 11. Assume that the desired rudder angle is represented by 6 bits plus a sign-bit to indicate port or starboard rudder. The 6 bits are fed into the left side of the *comparator*. The right side of the comparator is fed with the output of a shaft encoder which is mechanically coupled to the rudder itself. The comparator yields an output on one wire to indicate whether $A > B$ or $A \leqslant B$, and on the other wire to indicate whether $A = B$ or $A \neq B$. These outputs enter the *controller* and, with the sign information which gives the desired rudder quadrant, i.e., port or starboard, and the actual rudder quadrant as determined by the shaft encoder, the controller generates the appropriate signals to the step motor driver. The step motor turns both rudder and shaft encoder to make $A = B$ and when this occurs, the controller stops further stepping.

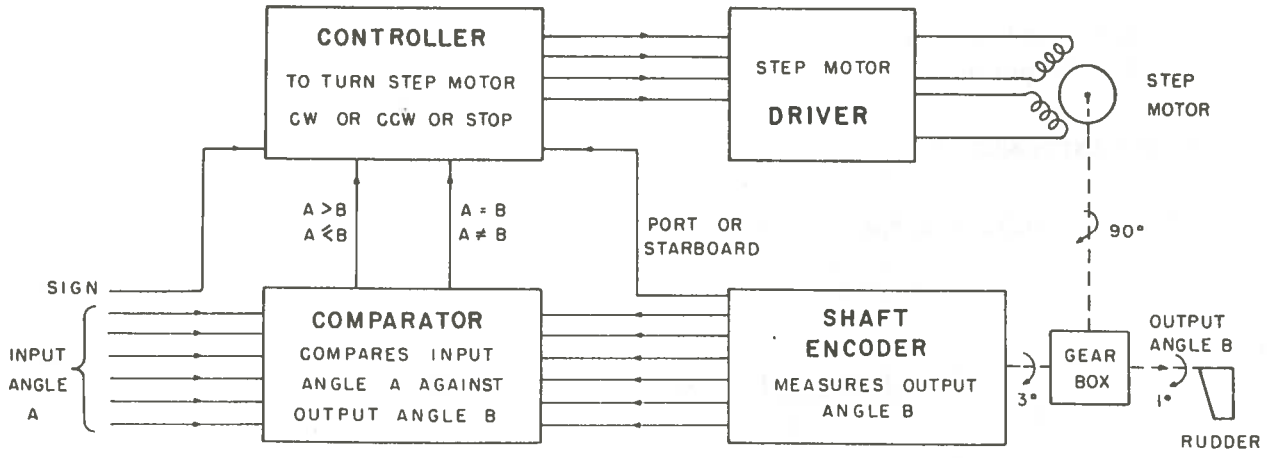


Figure 11 Block diagram of rudder positioner

Comparator

This is a 6-bit comparator which simply compares the magnitude of word A against the magnitude of word B. Word A consists of 6 bits, a_1 to a_6, and word B has 6 bits, b_1 to b_6. a_1 and b_1 are the most significant bits.

The corresponding bits, a_1 and b_1 are first checked. If a_1 is a ONE and b_1 is a ZERO, then this establishes that A is greater than B, since the lesser significant bits to the right of a_1 and b_1 may be ignored. If a_1 is equal to b_1, then a_2 is checked against b_2. If a_2 is a ONE and b_2 is a ZERO this makes A greater than B. There are many other possible combinations and these are listed in the table below as an example for 2 bits. This shows the 16 possible combinations of a_1, b_1, a_2, b_2, with H and G as the desired outputs:

a_1	0 1	0 1	0 1	0 1	0 1	0 1	0 1	0 1
b_1	0 0	1 1	0 0	1 1	0 0	1 1	0 0	1 1
a_2	0 0	0 0	1 1	1 1	0 0	0 0	1 1	1 1
b_2	0 0	0 0	0 0	0 0	1 1	1 1	1 1	1 1
H	0 1	0 0	1 1	0 1	0 1	0 0	0 1	0 0
G	0 1	1 0	1 1	1 1	1 1	1 1	0 1	1 0

where $H = 1$ indicates $A > B$
$H = 0$ indicates $A \leqslant B$

and $G = 1$ indicates $A \neq B$
$G = 0$ indicates $A = B$

The Boolean expression for H is by inspection

$$H = a_1\overline{b}_1\overline{a}_2\overline{b}_2 + \overline{a}_1\overline{b}_1a_2\overline{b}_2 + a_1\overline{b}_1a_2\overline{b}_2 + a_1b_1a_2\overline{b}_2 + a_1\overline{b}_1\overline{a}_2b_2 + a_1\overline{b}_1a_2b_2$$

This reduces to

$$H = a_1\overline{b}_1 + a_2\overline{b}_2(\overline{a}_1\overline{b}_1 + a_1b_1)$$

$$H = a_1\overline{b}_1 + a_2\overline{b}_2(\overline{C}_1)$$

where $\overline{C}_1 = \overline{a}_1\overline{b}_1 + a_1b_1$

Also G can be expressed as

$$\overline{G} = \overline{a}_1\overline{b}_1\overline{a}_2\overline{b}_2 + a_1b_1\overline{a}_2\overline{b}_2 + \overline{a}_1\overline{b}_1a_2b_2 + a_1b_1a_2b_2$$

$$= \overline{a}_2\overline{b}_2(\overline{C}_1) + a_2b_2(\overline{C}_1)$$

$$\overline{G} = \overline{C}_1(\overline{a}_2\overline{b}_2 + a_2b_2) = \overline{C}_1\overline{C}_2$$

where $\overline{C}_2 = \overline{a}_2\overline{b}_2 + a_2b_2$

If more than 2 bits are compared, it can be shown that

$$H = a_1\overline{b}_1 + a_2\overline{b}_2\overline{C}_1 + a_3\overline{b}_3\overline{C}_1\overline{C}_2 + a_4\overline{b}_4\overline{C}_1\overline{C}_2\overline{C}_3 + \ldots$$

and $\overline{G} = \overline{C}_1\overline{C}_2\overline{C}_3\overline{C}_4 \ldots$

The logic circuit to form H and G is shown in Fig. 12. Here, H is generated at the same time as G, making use of the partial G which is generated in 'succession'. No intentional delays are used.

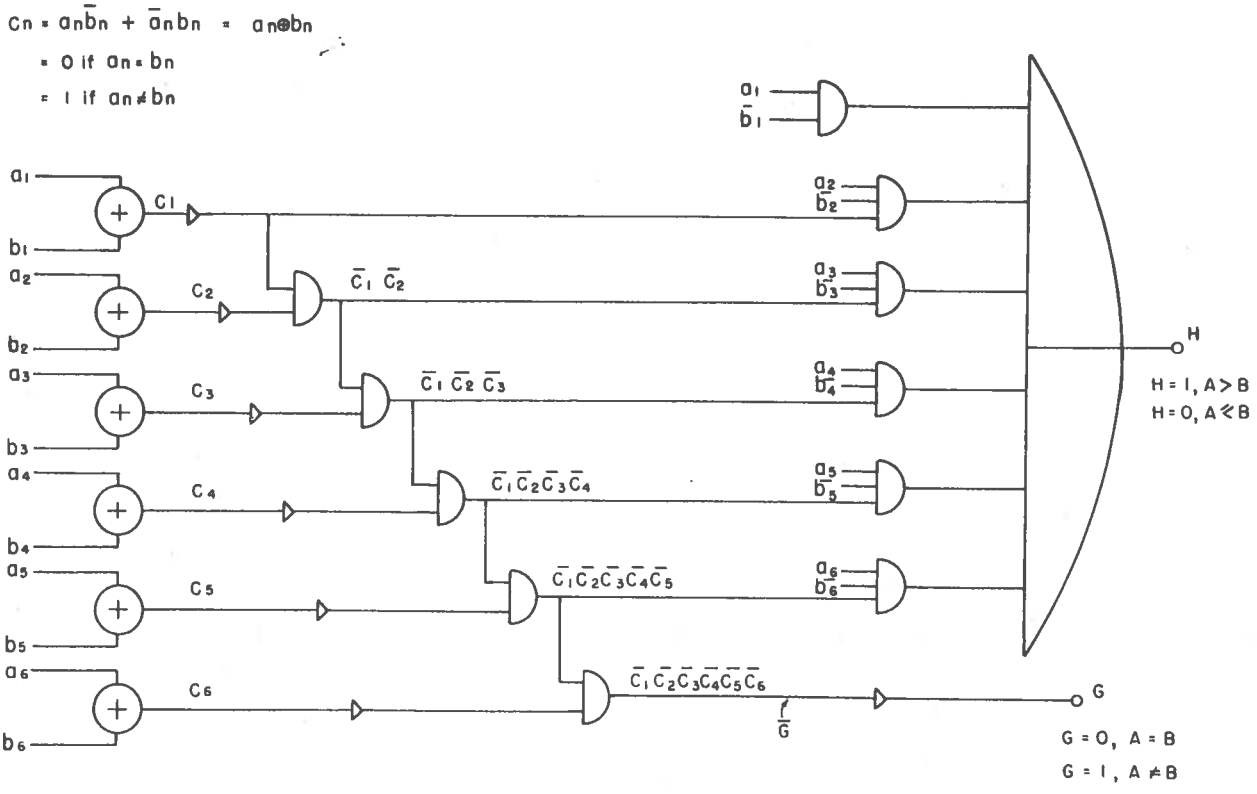


Figure 12 Six-bit comparator

Controller

The step motor is turned either clockwise (CW) or counter-clockwise (CCW) as long as there is an error between the output angle, B, and the input angle, A. When $A = B$, the controller stops the motor, provided that the rudder is in the requested quadrant.

Let $p = 1, 0$ represent the requested port or starboard rudder. The actual rudder position may be either port or starboard; let this be designated by $q = 1, 0$.

Recall that signals H and G were generated by the comparator. Consider the case when $G = 0$ $(A = B)$ and $p = q$ (rudder is resting in the requested quadrant). In this case only, the controller should stop further stepping of the step motor.

$$S = \text{stop stepping if } G = 0 \text{ and } p = q$$
$$= \overline{G}\,(\overline{p}\,\overline{q} + pq) = 1$$

In all other cases, the step motor must keep stepping. The direction of stepping is decided by whether $A > B$ or $A < B$ and also by p and q. For example, if $A < B$ and $p = 1 = q$, then the correct direction is CCW, as shown in Fig. 13.

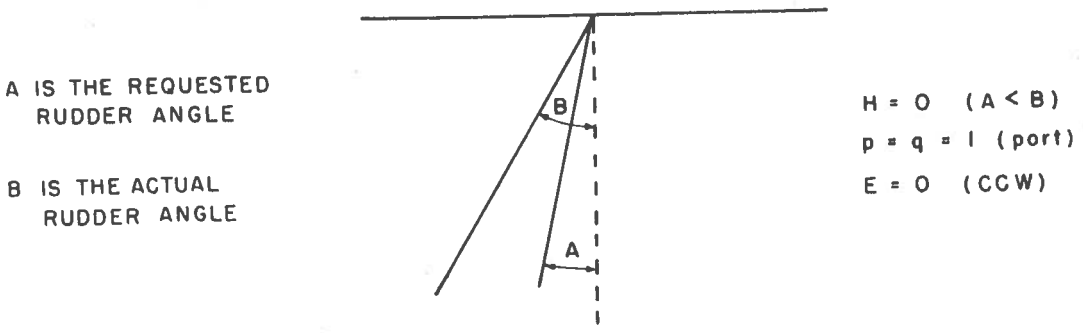


Figure 13 Rudder angles A and B

In Fig. 13, A is the requested angle. B is the desired output angle. Both are shown in the 'port' quadrant.

Evidently the rudder must move CCW to make $B = A$. The table below lists all possible combinations of the 3 variables H, p, and q. E is the desired output direction and this was obtained by inspection.

H	0	1	0	1	0	1	0	1
p	0	0	1	1	0	0	1	1
q	0	0	0	0	1	1	1	1
E	1	0	1	1	0	0	0	1

$H = 1, \quad A > B$
$H = 0, \quad A \leqslant B$

From this, we get $E = \bar{H}\bar{p}\bar{q} + \bar{H}p\bar{q} + Hp\bar{q} + Hpq$

which reduces to, $E = \bar{H}\bar{q} + Hp$

where, $E = 1, 0$ indicates either CW or CCW direction of turn of the step motor.

The generation of both S and E are shown in Fig. 14.

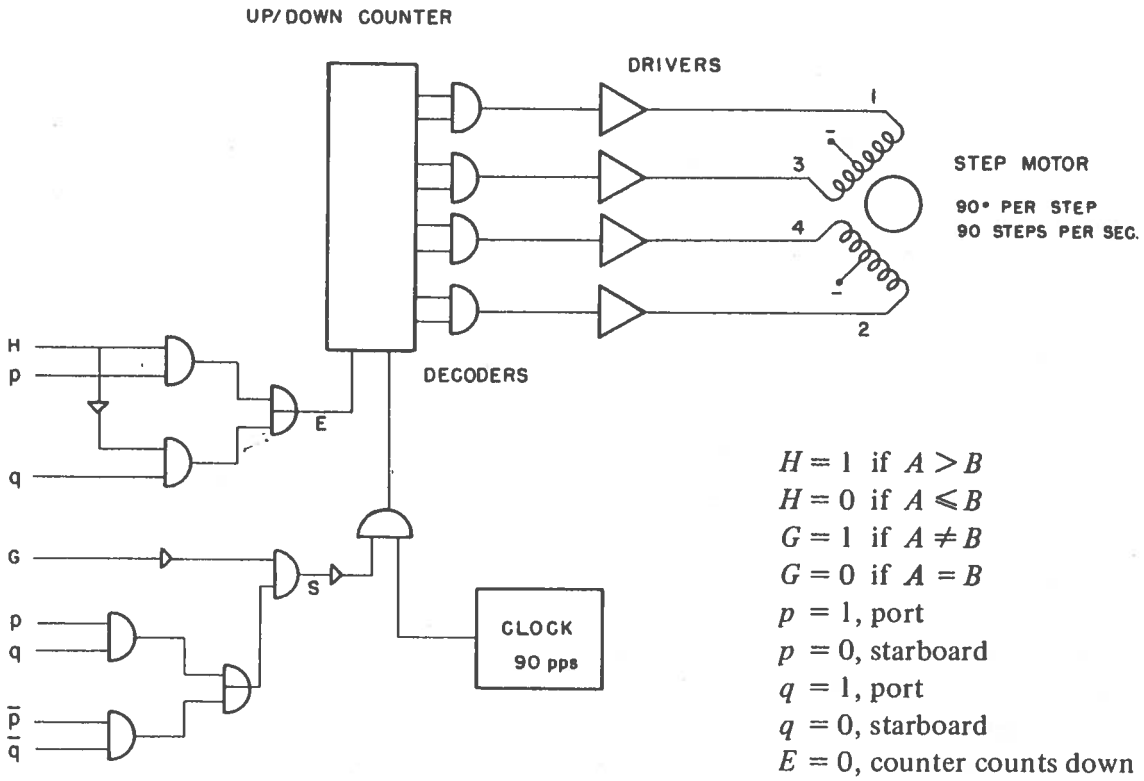


Figure 14 Controller

Further logic is necessary to provide suitable signals to the step motor driver. The stator windings of the step motor are pulsed sequentially in one direction for CW rotation of the rotor and pulsed sequentially in the other direction for CCW rotation of the rotor. If the winding terminals, labeled 1, 2, 3, and 4 in Fig. 14, are pulsed in that order, CW rotation results. If pulsed in the reverse order, CCW rotation results. This suggests an up-down counter using two flip-flops and a decoder to decode the four states. When $E = 0$ the counter counts down and when $E = 1$, the counter counts up. Counting is conditional upon S. When $S = 1$, counting stops. Details of this *controller* are shown in the supplement [1].

Gear Box and Shaft Encoder

The problem, as stated before, was to move the rudder in one-degree steps at a rate of 90 steps per second under an estimated reaction torque of one foot-pound. The final

accuracy was to be within $\pm\frac{1}{4}$ degree. Coarse steps of 5 degrees were not considered in the following solution as one degree steps would be equally acceptable.

Since the step angle was 90 degrees, one-degree steps were obtained simply by a 90:1 gear ratio between the stepper motor shaft and the rudder. With this ratio, load torque of 1 foot-pound is reflected as approximately 2 inch-ounces at the motor. The rotor must then hold its position to within $22\frac{1}{2}$ degrees so that the rudder position is held to $\pm\frac{1}{4}$ degree. In other words, a 2 inch-ounce torque must be developed at a rotor deflection angle of not more than $22\frac{1}{2}$ degrees. Maximum torque is developed at an angle of 90 degrees when the permanent magnet rotor is at right angles to the magnetic field. This leads to a simple method of calculating the required stall torque for the step motor.

$$\text{Stall torque required} = \frac{\text{Load Torque/Gear Ratio}}{\sin\ (\text{Gear Ratio} \times \frac{1}{4}\ \text{degree})}$$

$$= \frac{12 \times 16 \div 90}{\sin\ (90 \times \frac{1}{4})} = 5.6 \text{ inch-ounces}$$

(The variation between torque and angle is approximately sinusoidal).

Accordingly, a size 15 step motor (1 inch diameter, 2 inches long) with a peak torque of 7 inch-ounces and a maximum stepping rate of 165 steps/sec was selected to satisfy the static requirements.

Load inertia was considered but not in much detail. It suffices to mention that a mahogany rudder, 4 inches by 6 inches by $\frac{1}{2}$ inch thick and tapered away from the pivot point which was along the 6-inch side, had a moment of inertia of about $\frac{1}{3}$ gm cm^3 referred to the motor shaft. The rotor inertia was given as 5 gm cm^3 so that the rotational energy would be stored mostly in the rotor. The rotor itself lines up to a stiff magnetic field as evidenced by its ability to stop without overshooting. It was hoped that this, together with the damping action of the water would provide enough damping for the rudder.

The shaft angle encoder divides 360° into 120 sectors and each 3-degree sector is represented by a 6-bit Gray code plus a sign bit. This encoder is geared 3 : 1 to the rudder shaft so that a one-degree change in the rudder shaft causes a 3-degree change in the encoder. In this way each of the 120 positions of the rudder is represented by a distinct code. This gives a range of rudder movement of ± 60 degrees, although only ± 45 degrees was called for.

The Gray binary code (GB code) is a code in which only one bit is changed in going from one code word to the next. Because of this, exact mechanical alignment is not necessary in order to sense the change between adjacent code words. Further, since

the least significant bits alternate every two bits, the diameter of the encoder can be made one half the size of a pure binary (PB) coded disk. As an example, a 3-bit GB code is compared with a PB code.

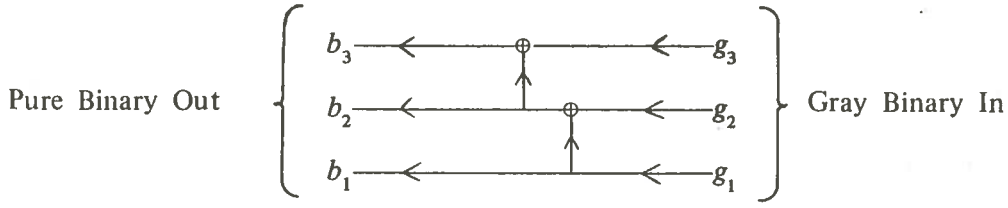
PB Code	GB Code
1 1 1	1 0 0
1 1 0	1 0 1
1 0 1	1 1 1
1 0 0	1 1 0
0 1 1	0 1 0
0 1 0	0 1 1
0 0 1	0 0 1
0 0 0	0 0 0
$b_1\, b_2\, b_3$	$g_1\, g_2\, g_3$

It is convenient to convert from the GB code to the PB code so that the comparator will compare in the same code. By inspection, it is evident that the most significant bit is the same in either code, i.e., $b_1 = g_1$. The next bit, b_2, is a ONE if g_1 and g_2 are odd, and a ZERO if even. Similarly, b_3 indicates the oddness or evenness over the 3 bits g_1, g_2, and g_3.

Logically, these statements reduce to,

$$b_n = b_{n-1}\,\overline{g}_n \;+\; \overline{b}_{n-1}\,g_n$$

where, b_n is the nth PB code bit and g_n is the nth GB code bit, i.e., b_3 is a ONE if either b_2 or g_3 is a ONE, but not both. Let the symbol $\oplus$ designate this. Conversion for 3 bits is shown below.



The b's enter the comparator to be compared against the a's which represent the desired rudder angle. This completes the loop.

The complete logical circuitry is shown in the supplement [1].

Motor Speed Control

A number of different methods are available for controlling speeds of dc motors, ranging from purely digital to analog methods. A hybrid scheme was suggested by the

problem itself. The problem was to control three motors independently. The motors were specified by the user as $\frac{1}{3}$ hp dc motors because these were on hand from previous work. The motors were to be driven from a 110-volt, 60-cycle gas-electric generator. The desired speed control range was approximately 50 to 1000 rpm with an accuracy of about ± 5 rpm under steady load conditions. Fast response to abrupt load or input speed changes was not required. Reversing, starting and stopping were additional controls required.

The solution took the following form (see Fig. 15). (*a*) Measure the actual shaft speed with a toothed-wheel and a counter; (*b*) Compare this with the demanded speed; (*c*) Drive a step motor to change the triggering point of SCR's; (*d*) Use SCR's and rectifiers in a bridge arrangement for full-wave, reversible operation of the motor.

Digital sensing of motor speed and digital comparison of this against the demanded input speed which was also digital, was suggested in order to get long term stability, i.e., drift-free and calibration-free operation. The important thing was to maintain known propeller speeds from one test run to another which could occupy a span of a few minutes up to a few days. Repeatability was important.

No attempt was made to scale the propulsion characteristics down to the model ship size. Constant propeller speed was chosen instead of constant horsepower because this would be easier to implement and the results easier to analyze.

Use of a step motor to change the phase angle and hence the firing point on the ac waveform to the SCR's had the effect of integrating the error, thus eventually reducing the steady-state speed error. Also it was a convenient component to interface the digital error signals with the analog phase changing part of the system. However, this convenience was bought at some cost – circuit complexity and poor transient response.

Use of SCR's and rectifiers permitted full-wave operation and a fully solid-state reversing scheme. In addition, simple binary control signals into the appropriate points in the circuitry made starting/stopping or forward/reverse functions quite easy to implement. The following is a description of the parts which make up Fig. 15. Further circuit details are included in the supplement [1].

Measurement of Shaft Speed

Shaft speed was measured in the following way. A toothed-wheel and a light-source/photo-electric pick-off arrangement produced pulses which were gated into a counter. A 192-tooth gear and a $\frac{1}{64}$th second gate were chosen to give a weight of 10 rpm per pulse. The light source consisted of two bulbs arranged electronically so that failure of one would automatically switch in the other. This was done as a precaution against run-away speeds when feedback was lost. The pick-off electronics consisted of an amplifier followed by trigger circuits to produce pulses on the leading and trailing edges of each tooth. This was followed by an 8-bit pure binary counter. Eight bits were required to take care of 'overflow' even though the input speed was represented by 6 bits.

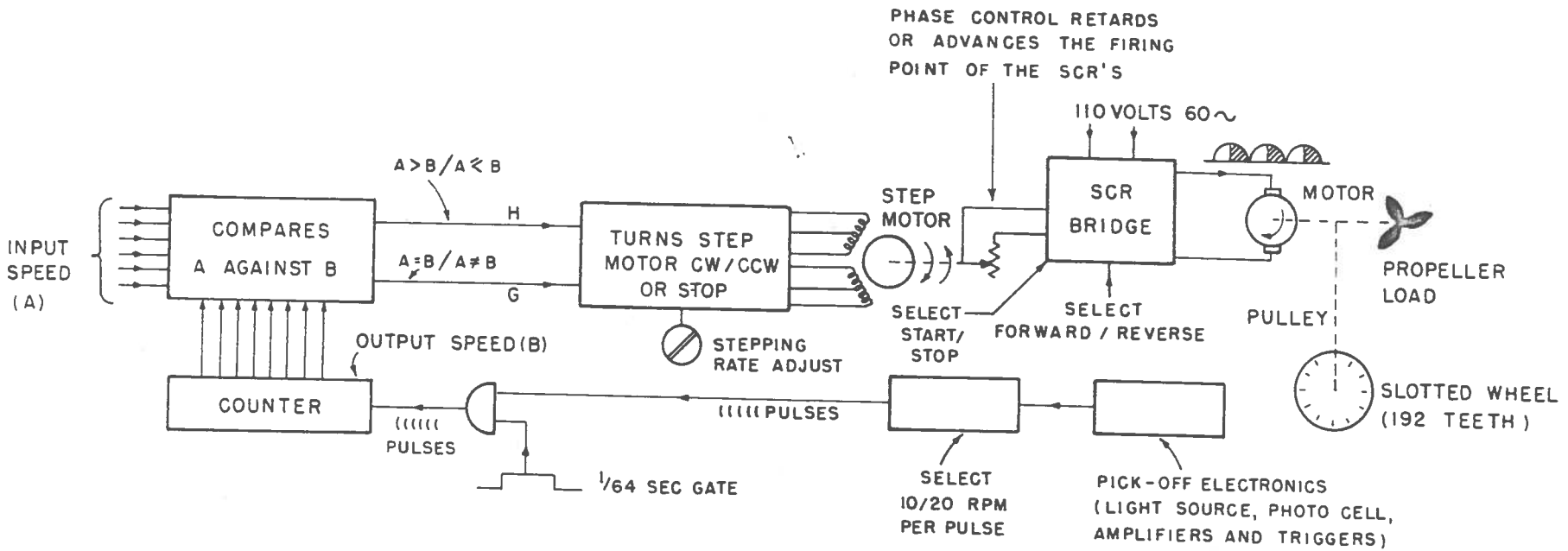


Figure 15 Propeller speed control

Ten- or 20-rpm increments were selected on command. Top speed for the 10-rpm increment was 630 rpm and for the 20 rpm increment, 1260 rpm. This was done simply by inhibiting every other pulse to produce, effectively, a 'weight' of 20 rpm for the least significant bit.

Comparison of Input Command Speed With the Output Shaft Speed

This comparison was done digitally and the results of the comparison were simple binary signals on two output wires. From these signals it was possible to resolve whether the input commanded speed was greater or less than, or equal to the actual output shaft speed.

Let A represent the input commanded speed and B, the output speed. Three possibilities arise: $A > B$, $A < B$, or $A = B$. One output wire yields a binary signal indicating $A > B$ or $A \leqslant B$ and the other output wire gives a signal indicating $A = B$ or $A \neq B$. The situation $A \leqslant B$ is resolved by the second wire. These signals are further interpreted by the step motor controller as CW/CCW or STOP/GO signals. If $A > B$, the step motor goes CCW to increase the firing angle and if $A < B$, the step motor turns CW to decrease the firing angle. When $A = B$, the step motor stops.

Comparator Logic

The comparator checks to see whether A is larger than or equal to B. The comparison starts by looking at the most significant bit of word A and the MSB of word B. Let these two bits be represented by a_1 and b_1 (a's and b's are just the coefficients). If a_1 is larger than b_1, i.e., $a_1 = 1$, and $b_1 = 0$, then A is larger than B and no further checks are necessary. If a_1 is not larger than b_1 then either $a_1 < b_1$ or $a_1 = b_1$. If $a_1 = b_1$ a check is made on the next MSB's of A and B, and so on.

These statements can be written in a symbolic form where $H = 1$ means $A > B$ and $H = 0$ means $A \leqslant B$. Hence

$$H = a_1\overline{b}_1 + (\overline{a}_1\overline{b}_1 + a_1b_1)\,a_2\overline{b}_2 + (\overline{a}_1\overline{b}_1 + a_1b_1)\,(\overline{a}_2\overline{b}_2 + a_2b_2)\,a_3\overline{b}_3 + \ldots +$$

where the first term produces a 1 if $a_1 = 1$ *and* $b_1 = 0$ *or* the second term yields a 1 if $a_2 = 1$ *and* $b_2 = 0$, provided $a_1 = b_1$ and so on.

Letting $C_n = \overline{a}_n b_n + a_n\overline{b}_n$ or $\overline{C}_n = \overline{a}_n\overline{b}_n + a_n b_n$

$$H = a_1\overline{b}_1 + \overline{C}_1 a_2\overline{b}_2 + \overline{C}_1\,\overline{C}_2 a_3\overline{b}_3 + \ldots$$

$$+ \overline{C}_1\overline{C}_2\overline{C}_3\overline{C}_4\overline{C}_5 a_6\overline{b}_6$$

The situation where $A = B$ can be sensed by combining all the C's, including C_6, to form

$$\overline{G} = \overline{C}_1 \overline{C}_2 \overline{C}_3 \overline{C}_4 \overline{C}_5 \overline{C}_6$$

where $G = 0$, 1 represents $A = B$, $A \neq B$.

Both H and G are formed by a circuit similar to Fig. 2 which was used for rudder control.

Stepper Motor Controller

The two signals H and G enter a controller which produces CW or CCW stepping pulses in accordance with H and stops the stepping pulses in accordance with G.

The controller consists of two flip-flops connected as an up/down counter and fed from a gate through which stepping pulses pass if $G = 1$. ($A \neq B$).

Two flip-flops with simple decoding provide the necessary signals to energize the stator windings on the step motor to produce either CW or CCW rotation. The basic circuit to do this is shown in Fig. 14.

Phase Shifter and SCR Circuit

The phase shift circuit consists of a resistor in series with a capacitor, connected across a center-tapped secondary winding of a transformer. The output voltage appears across the junction of R and C to the center-tap. Changing R changes the phase of the output voltage with respect to the input. Maximum range is 180 degrees for $R = 0$ to $R = \infty$.

The zero crossings of the output voltage are detected and pulses produced at these points to control the 'firing angle' of the SCR's. These pulses enter at the lower left of Fig. 16.

This circuit (Fig. 16) must perform the following functions:

(*a*) Start and stop motor
(*b*) Control the speed of the motor
(*c*) Reverse the motor

The circuit consists essentially of three bridges composed of recitifiers and SCR's. Bridge *H*, *G*, *I*, and *J* supplies full-wave dc for the field. Bridge *H*, *G*, *A*, and *B* supplies variable full-wave power to the motor bridge. Bridge *C*, *D*, *E*, and *F* (motor bridge) controls the direction of current in the armature to make it go forward or reverse.

If the commands are START and FORWARD, SCR's *A*, *B*, *E*, and *D* are all fired simultaneously and current goes down the armature to drive the motor in the forward direction at the required speed.

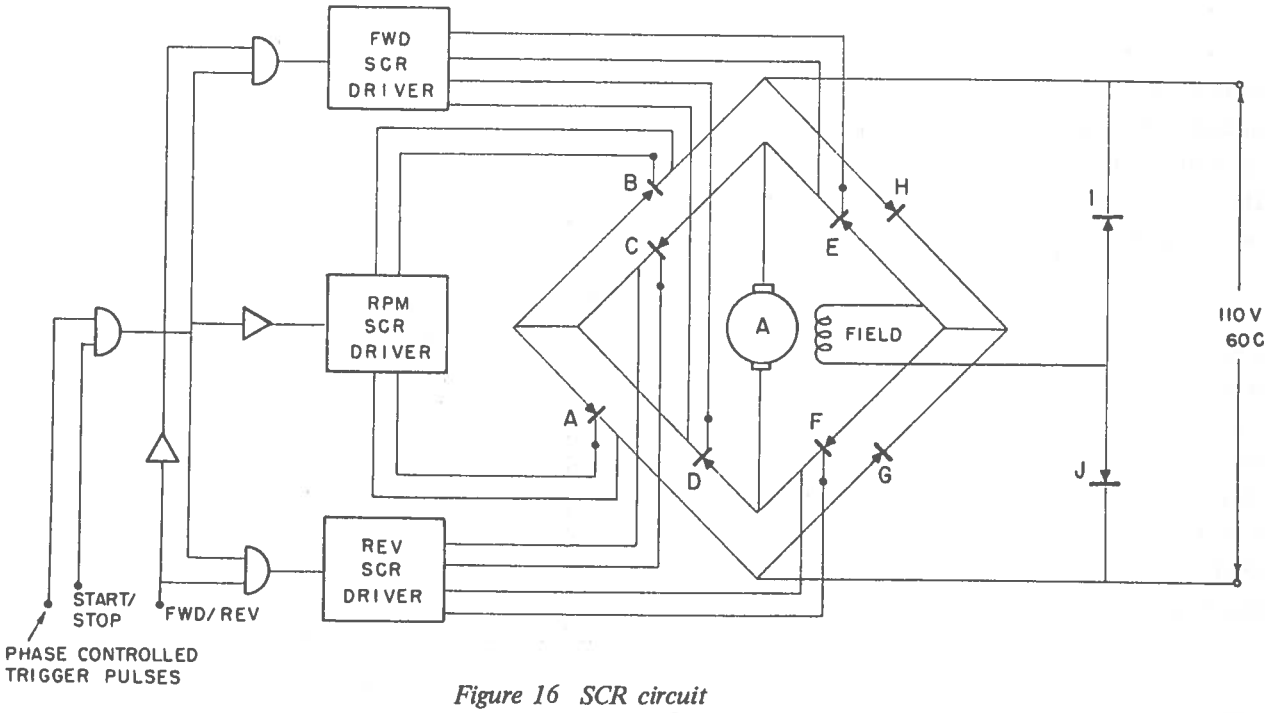


Figure 16 SCR circuit

If the commands are START and REVERSE, SCR's, *A*, *B*, *C* and *F* are fired simultaneously and current goes up the armature to drive the motor in the reverse direction at the required speed.

If the STOP command is given, no SCR's are fired and the motor stops.

Time Sharing of the Comparator

Even though the comparator resolves a simple problem (is *A* equal to *B* and if not, which is larger?) the circuitry consists of many gates. For this reason one comparator was shared by the three motor-speed controllers. This was done by multiplexing the input speed information – say, A_1, A_2, and A_3 for motors 1, 2 and 3 – into one input of the comparator and then multiplexing the feed-back speeds B_1, B_2, and B_3 into the other input.

The results of each comparison were stored in flip-flops. From these flip-flops each step motor controller received information to turn the step motor CW/CCW or STOP. Two bits are required for each comparison and, therefore, six flip-flops were used for storing the three *H*'s and the three *G*'s. The comparator which was used for rudder position control was not multiplexed, although in principle this was possible.

Stability Problems

When either the propeller load or the input commanded speed changed abruptly, there was overshoot and a long settling time. This depended on the gain setting as well as on the size of the sudden load or input change.

For different propellers (3 blades, 5 blades) and different shaft loads (whether a gear box was used or not) the servo responded differently. Consequently, an on-site gain adjustment was necessary to prevent oscillations or instability. With low gain, it took the system longer to reach the demanded speed but the overshoot was less. With higher gains, the system reached the demanded speed more quickly but settled more slowly – after two or three oscillations. Gain was adjusted by changing the stepping rate of the step motor.

For uniform and slow changing loads and for slow input command speed changes, the response was satisfactory. The 'steady-state' error was small because the step motor acted as an error integrator. So, as long as there was any difference between input and output speeds, the system continually integrated this and corrected the error. Under uniform load conditions, the mean value of this difference or error was close to zero, as one would expect for this type of servo (error integrated) but a surprising result was the low (2.7 rpm, rms) value of the fluctuations. This remains to be explained – a guess is that a certain amount of 'dither' in the feedback speed unintentionally reduced the fluctuations.

Output shaft speed was sampled 32 times per second and the time required for a measurement (gate time) was $\frac{1}{64}$ sec. The comparator gave results 32 times a second for each shaft. Step motor rate was adjustable between 0 and 100 steps/sec with about 30 steps/sec giving reasonable results.

Much more needs to be done to improve the transient response of the speed servo by suitable analysis rather than the cut-and-try approach that was taken.

Latching Relay Control

Since a variety of ON/OFF output devices were specified by the user, a set of 28 push-buttons were provided. As described earlier, 14 of these were labeled *A* to *N* and the other 14 labeled from 1 to 14. An output selection was made by pressing two buttons (either simultaneously or in any order) a *lettered* button and a *numbered* button. This produced two code words which were decoded at the receiving end. A latching relay was set when both code words were received correctly. For example, *A5* would set the relay, and *A6* would reset the relay.

As well as turning ON or OFF certain output devices during a test, the relays were useful in 'holding' OFF-type instructions. This prevented spurious results during the starting of the gas-electric generator.

Tests and Results on the Remote-Control System

The receiving end of the system was installed in an 18-foot model tanker. This gave enough room for various test instruments and a cock-pit for a man on board. The transmitting end was put on a 35-foot tower on one side of the turning basin.

Most of the routine checks were made while the model was dry-docked; after these, a man was put on board to monitor the key signals while the equipment was under radio control (and to abort the remote control if necessary).

Commands

All commands (binary control signals) were displayed on indicator lamps. These were monitored for steadiness as well as the correctness of the code. Any flickering or fluctuations of these indicator lamps indicated a fault; this happened in low-signal areas of the tank when the transmitter power was reduced intentionally as described later.

Receiver Output

The IF output after detection was monitored by an oscilloscope. The peak-to-peak voltage of this video signal and the rise-and-fall times of the signal were observed for different areas of the tank and for different transmitter powers. In most of the working area the receiver output was saturated when the full transmitter power was used. The output came out of saturation when the model was near the wave-maker.

Frame Sync Pulse

A count was made of the number of sync pulses extracted by the sync detector. Under low-signal conditions the count dropped, this drop in count occurring suddenly as the signal level fell below a pre-set threshold, and then came up suddenly to the normal count as soon as the signal exceeded the threshold. This was normal.

Threshold (Slicing level of video)

This was adjusted above the detector noise to minimize false synchronizing and erratic operation when the signal was low or in the event of transmitter failure. The detector noise referred to above had the usual Rayleigh distribution for an envelope detector; the threshold was set higher than the peaks of this noise so that the ZEROS (indicated by the absence of carrier) would not be contrued as ONES. This setting was made once and as it turned out, was not as critical as originally anticipated.

Transmitter Power

The amount of transmitter power needed was estimated initially by calculation. A 50-mW transmitter was used; this was successively reduced in power by means of an attenuator to determine the minimum 'safe' power. The worst area (nulls caused by the metal wave-making machine) was along one end of the turning basin and, accordingly, the model ship was manoeuvred under radio control (but with a man on board) and the key signals were again monitored as the transmitter power was reduced. It was determined

that a reduction in power to 1/100 could be tolerated in the worst areas, and certainly a reduction to 1/10 power gave a good margin of safety.

Motor Speed and Rudder Position Servos

The stability of the speed servo and the position servo was observed during these tests. Initial adjustments were made while dry-docked; the stepping rates of the stepper-motors were adjusted to prevent large overshoots and to settle within a few cycles of (damped) oscillations. Then, with the model in water, the servos were checked to see that they settled while under load.

Under a steady load, the accuracy of the speed servo was checked; these were rather coarse checks using an electrical tachometer with a dial, and a mechanical tachometer much like a stop-watch. Both gave normal values, and from these readings it was easy to confirm that the speed commands were being obeyed. A change of 10 rpm was easily read but the small fluctuations in speed were not easily read.

Rudder positioning accuracy had already been checked previously in the laboratory under a static load; this gave $\frac{1}{3}$-degree deflection under a 1 foot-pound load. No further measurements were made on this, but a pointer attached to the rudder shaft and a graduated scale gave adequate indication that the rudder shaft followed the rudder commands.

Other remote commands, such as FORWARD or REVERSE, LOW RANGE (X 1) or TWICE the LOW RANGE (X 2), and START or STOP were checked. Also a FAST SLEW or SLOW SLEW mode on the rudder servo was selected by a remote command.

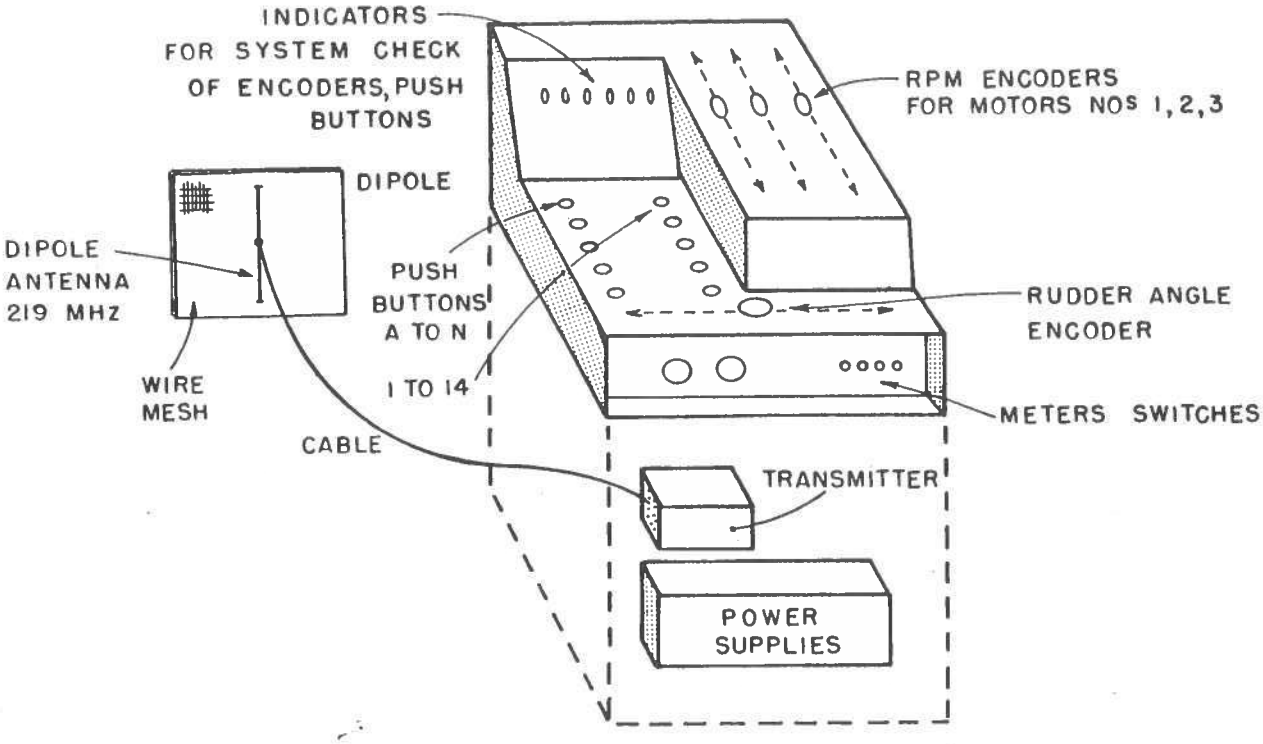
ON/OFF Commands

The tests on the ON/OFF-type commands were simple and consisted of observing the opening or closing of latching relays in accordance with the commands.

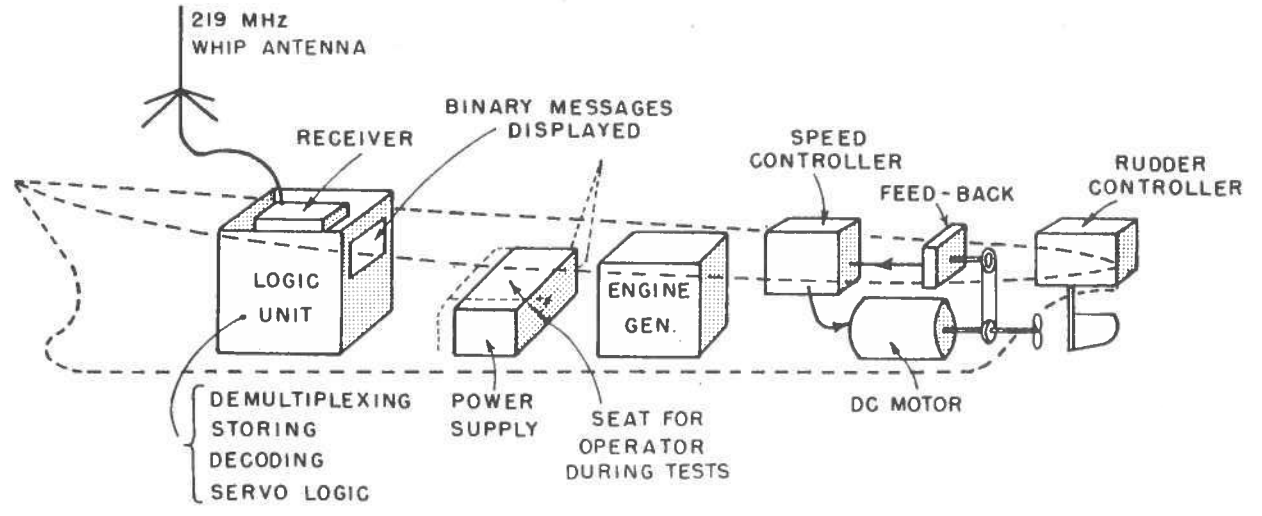
Modifications

During these tests it was found that a definite START-UP procedure and a SHUT-DOWN procedure was advisable; simple procedures were established to insure against false starts.

After the above tests, when we had become familiar with the system, the model tanker was remotely controlled in waves. Later, a model Catamaran was fitted with the control equipment. Two rudders were tied with a linkage system and driven by the rudder servo. Two propulsion motors were controlled; each motor was independently controlled in speed. In this case, the FORWARD/REVERSE selection was done by push-button commands for better reliability.



(a) Transmitter end



(b) Receiver end

Figure 17 Remote control equipment

Conclusions

Based on the results so far, the digital remote control system appears to have the necessary accuracy even after long periods of non-use. It is drift-free because of the digital form of the signals and as long as the RF link is solid, i.e., the received signals are above the threshold, the commands are recognized reliably.

Protection against impulse noise appears to be adequate by a simple parity check on each received command and further, by using a wideband receiver, the impulse noise is preserved as a narrow pulse, occupying just a small fraction of the bit time.

The simple modulation/demodulation scheme has proved to be workable for this job without unduly high transmitter power and with a simple demodulator (envelope detector) at the receiving end.

Installation requires careful attention to small details – mainly electrical and mechanical connections – which are easily overlooked during the initial fitting. These, of course, eventually cause malfunctioning of some part of the control system. Tracing back from the malfunction (symptom) to the cause proved to be tedious; it was difficult to recognize the symptoms. Usually the half-split technique of partitioning the location of the possible fault was resorted to; this proved to be most useful for the less obvious cases. Digital indicators were used on both transmitter and receiver.

More running time of the system on different model ships would give more data on its reliability, serviceability, and so on. So far, two model ships have been instrumented with the remote-control gear with satisfactory results, demonstrating the feasibility of a digital remote-control system for testing model ships.

Reference

1. Ayukawa, K., Foster, W.T., and Vachon, F. Remote control of model ships, Circuit diagram supplement. NRC Rept. ERB-822. 1969.

Acknowledgments

The authors wish to acknowledge the assistance given by members of the Ship Laboratory, particularly Mr. R.J. Peters. We also thank H.R. Smyth and S.A. Stone of this Division for their support of the project.

Bibliography

Watts S. Humphrey, Jr. Switching circuits with computer applications. McGraw-Hill Book Company, Inc., New York, 1958.

R.S. Ledley. Digital computer and control engineering. McGraw-Hill Book Company, Inc., New York, 1960.

S.W. Golomb, et al. Digital communications with space applications. Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1964.

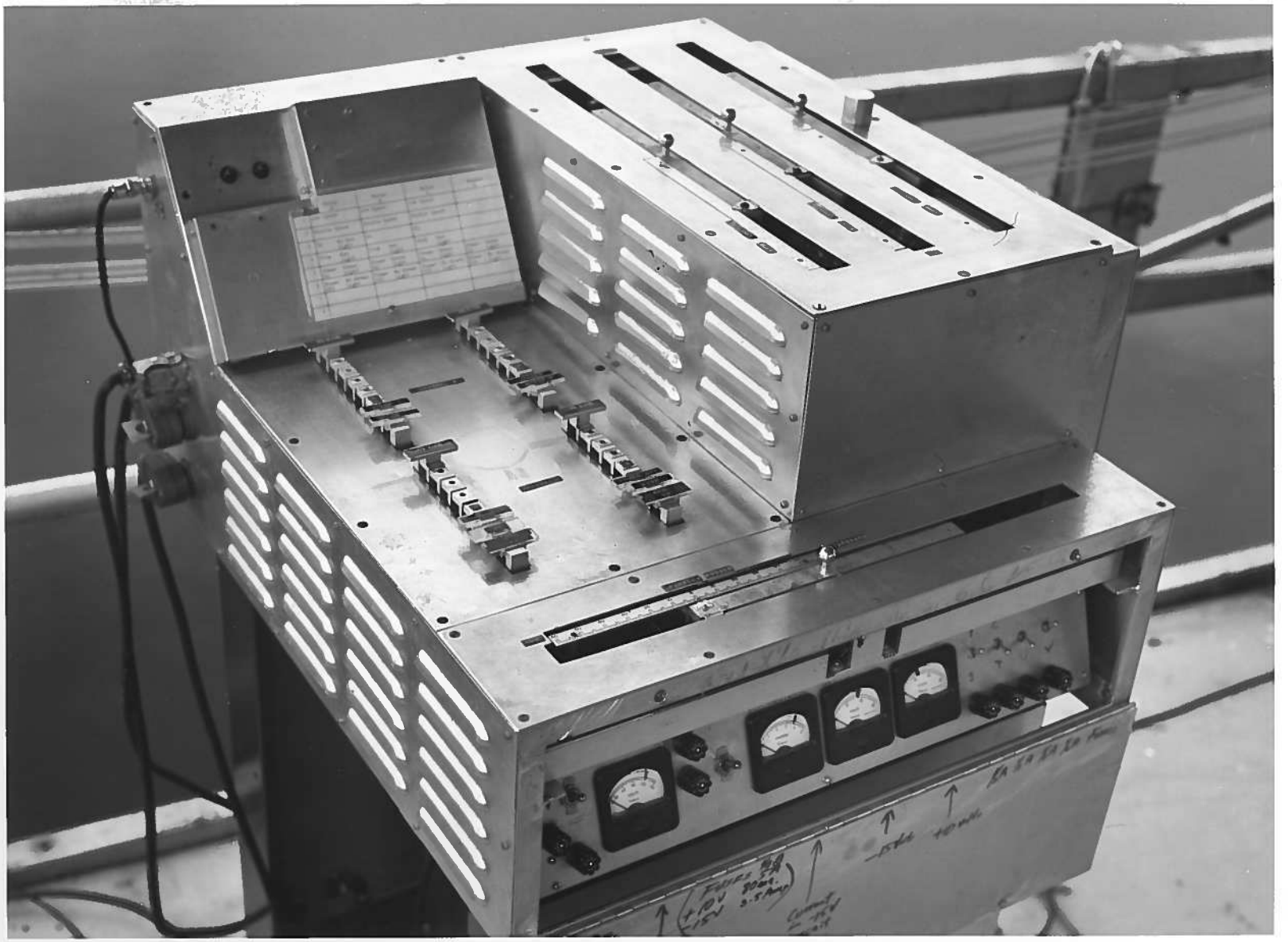


Plate I Tower equipment

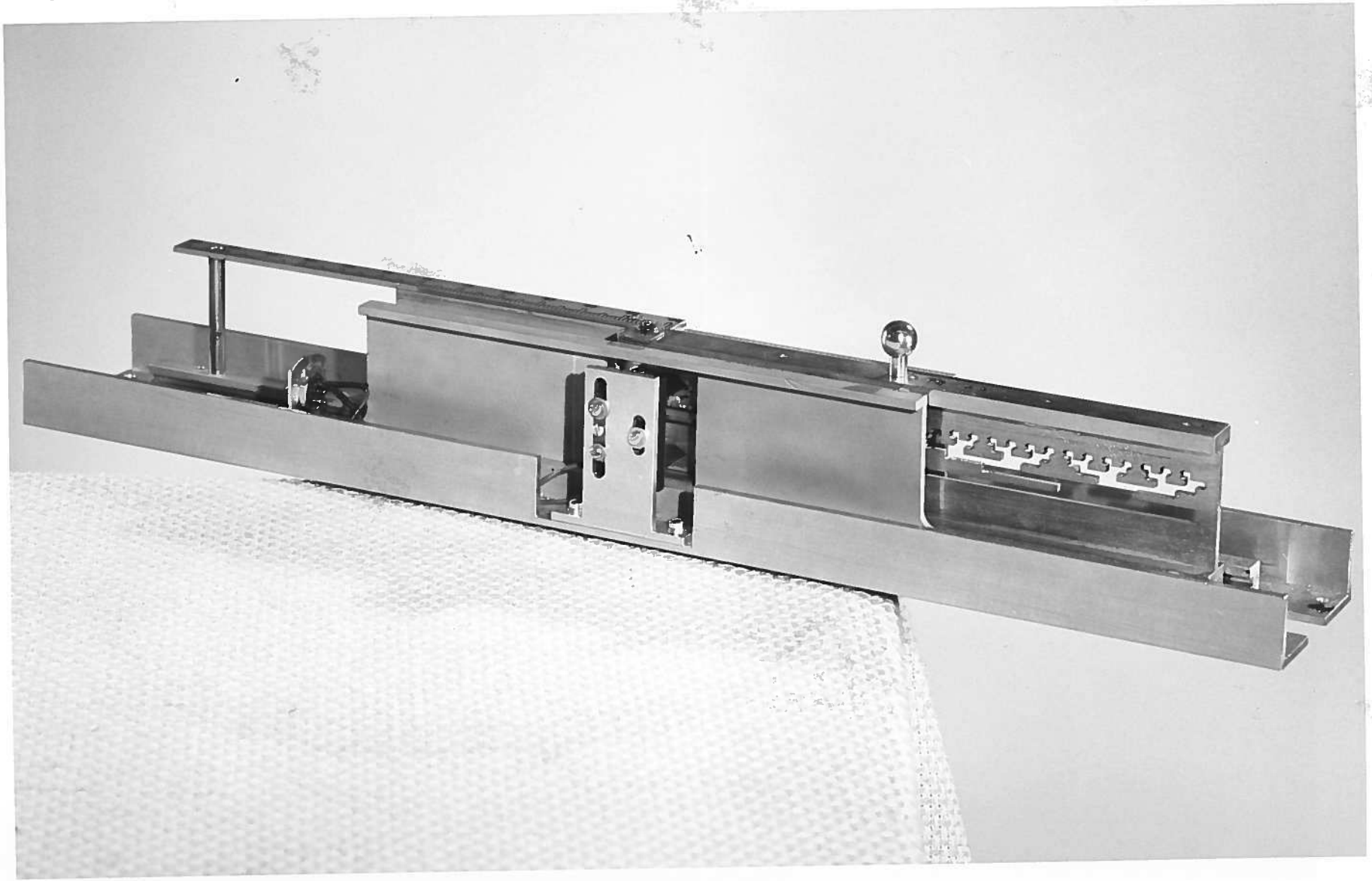


Plate II An encoder

Plate III Receiving end — digital type