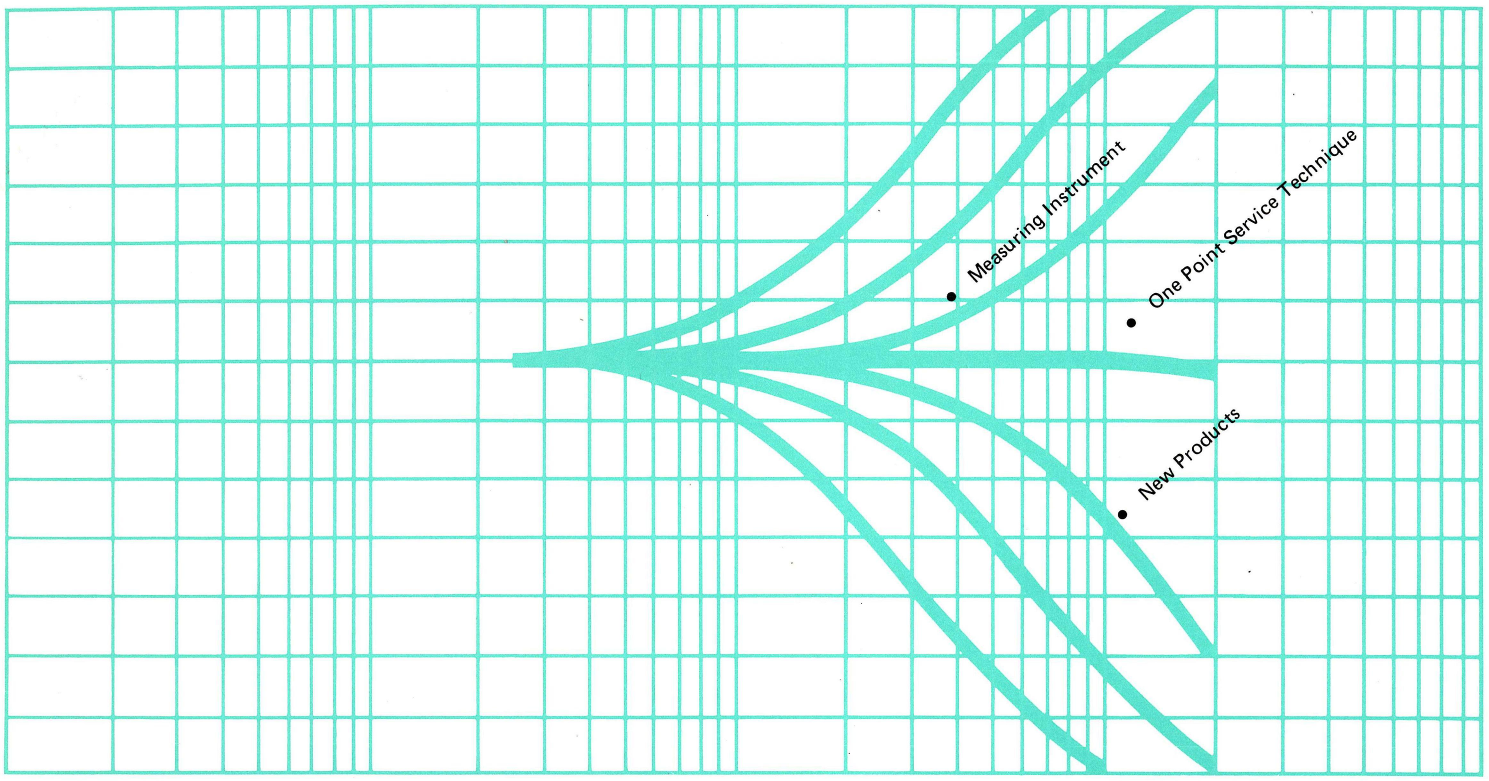
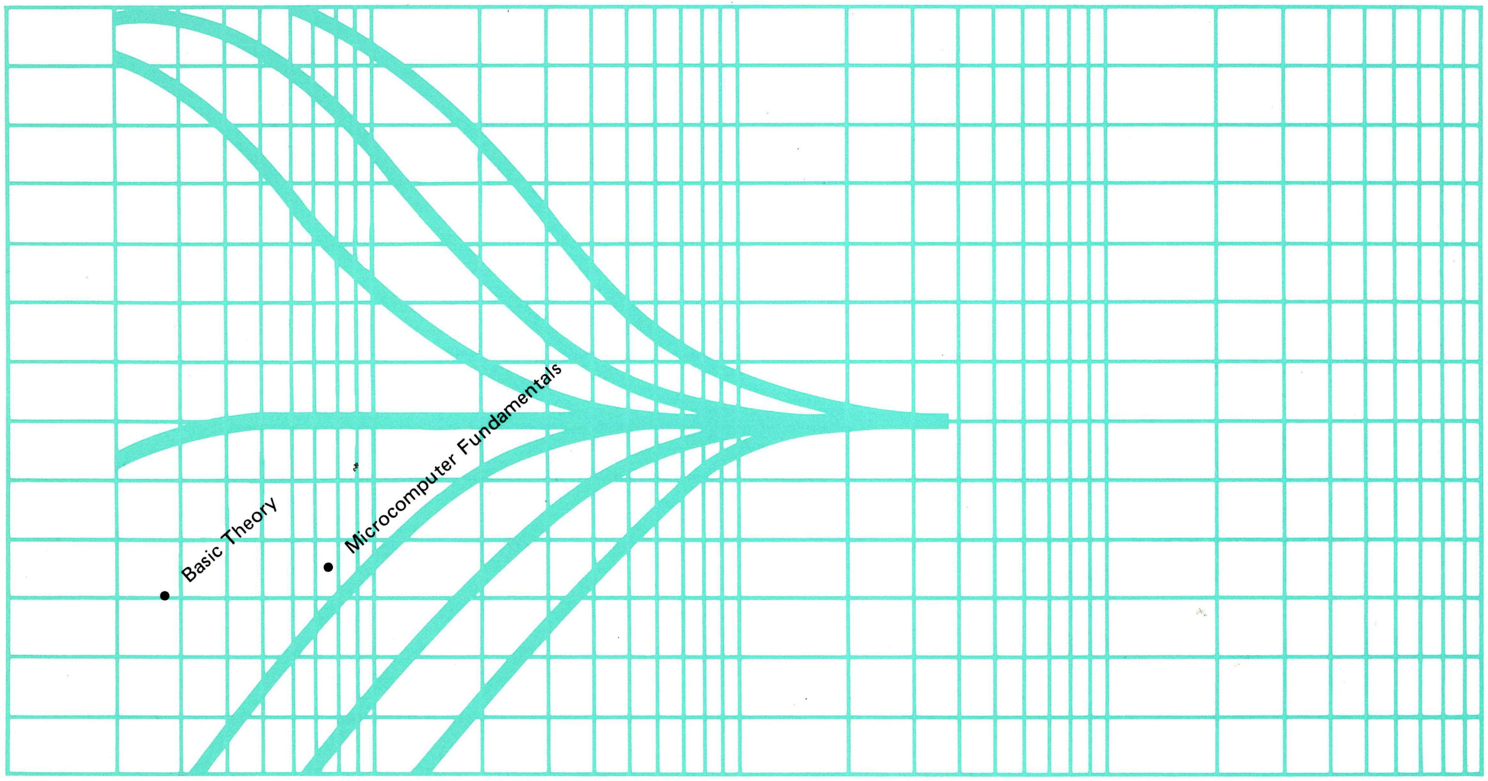


TUNING FORK

Audio and Video Service Guide

No. 9



PIONEER®

ORDER No. GGF-609

CONTENTS

Basic Theory

Transistor Circuits

There are two types in transistors, NPN and PNP types as shown in Fig. 1. The NPN type is made of silicon. PNP transistors have been made of germanium. The germanium transistors, however, are being superseded by silicon transistors. A transistor has three main electrodes; emitter (E), base (B) and collector (C).

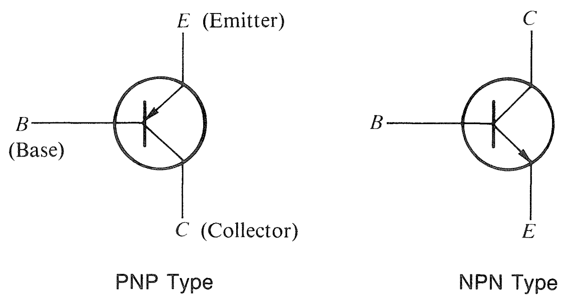


Fig. 1 Transistor Symbols

1. Typical Transistor Circuits

In wiring a transistor there are three main methods; common-emitter, common-base and common-collector wiring.

(a) Common-emitter Amplifier

In Fig. 2 the emitter element is common to both the input and output circuits. The amplification of this circuit is high. Its input impedance is medium and output impedance is high. This circuit is the most popular in amplifiers. It gives the following equations:

$I_E = I_C + I_B \qquad h_{FE} = I_C/I_B$

h_{FE}: DC current amplification with emitter grounded

The voltage amplification (A_V) is as follows:

$$A_V = \frac{V_O}{V_I} = \frac{-R_L \cdot I_C}{R_I \cdot I_B} = -\frac{R_L \cdot h_{FE} \cdot I_B}{R_I \cdot I_B} = -\frac{R_L}{R_I} h_{FE}$$

R_I: Input impedance
R_L: Load impedance

The voltage amplification can be known by dividing output impedance by input impedance and then multiplying the result by h_{FE}. The minus mark means that the input phase is inverted at the output.

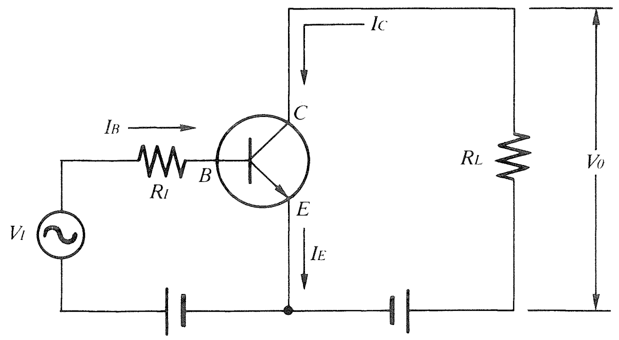


Fig. 2 Common-emitter Amplifier

(b) Common-base Amplifier

The features of this circuit are low input impedance and high output impedance. The current amplification (h_{FB}) of this circuit is:

$h_{FB} = (I_C/I_E) < 1$

h_{FB}: DC amplification with base grounded

The voltage amplification (A_V) is:

$$A_V = \frac{V_O}{V_I} = \frac{R_L \cdot I_C}{R_I \cdot I_E} = \frac{R_L \cdot h_{FB} \cdot I_E}{R_I \cdot I_E} = \frac{R_L}{R_I} h_{FB}$$

R_L/R_I should be made large to increase the amplification because h_{FB} is less than 1. The input and output signals are inphase each other.

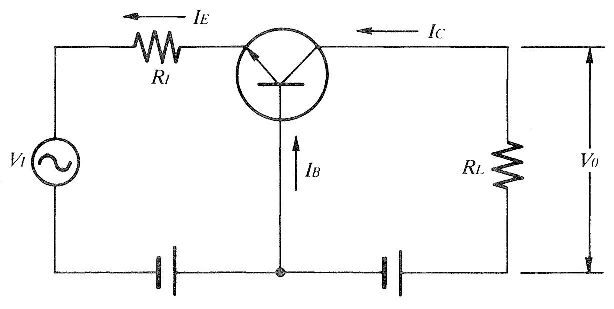


Fig. 3 Common-base Amplifier

(c) Common-collector Amplifier

This is also called emitter-follower. In this circuit current amplification is high, voltage amplification is less than 1, input impedance is high and output impedance is low. This is, therefore, used as a buffer.

Buffer: A circuit which converts impedance and isolates one electrical circuit from another to prevent undesirable interaction between two stages.

This gives the following equations.

$$I_E = I_C + I_B = h_{FE}\,I_B + I_B = (1+h_{FE})\,I_B$$

The input impedance (Z_I) becomes nearly the multiplied value of R_E by h_{FE}:

$$Z_I = \frac{(1+h_{FE})\,I_B R_E}{I_B} \fallingdotseq h_{FE}R_E$$

The output impedance (Z_O) is:

$$Z_O = \frac{R_I\ I_C/(1+h_{FE})}{I_C} \fallingdotseq \frac{R_I}{h_{FE}}$$

A_v is nearly equal to 1 because R_E is generally much larger than R_I. The output phase is the same as input phase.

$$A_V = \frac{V_O}{V_I} = \frac{R_E I_B(1+h_{FE})}{R_I + R_E I_B(1+h_{FE})}$$

$$= \frac{R_E(1+h_{FE})}{R_I + R_E(1+h_{FE})} \fallingdotseq 1$$

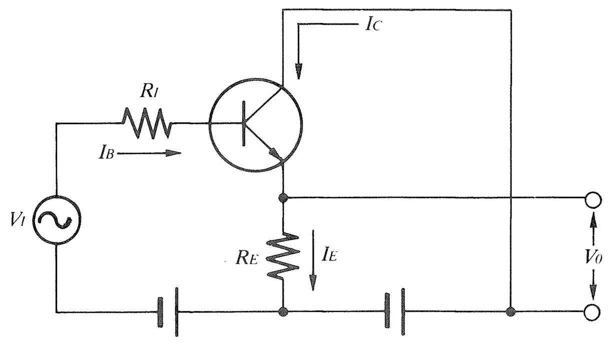


Fig. 4 Common-collector Amplifier

Table 1 Typical Transistor Amplifiers

Amplifiers	Common-emitter amp.	Common-collector amp.	Common-base amp.
Input impedance	Medium (1.95k)	High (300k)	Low (13)
Output impedance	High (1k)	Low (19.7)	High (1k)
Voltage amplification	−75.6	1	75.6
Current amplification	−150	150	1
Usage	Signal	Buffer	Buffer

2. Biasing

To operate a transistor, a DC voltage should be applied between the base and emitter in addition to the collector and emitter as shown in Fig. 5. The voltage between the base and emitter (V_{BE}) is called "bias voltage". The collector current flows when V_{BE} exceeds approximately 0.6V in silicon transistors (0.2V in gelmanium transistors).

$$V_{CE} = V_{CC} - I_C\,R_L$$

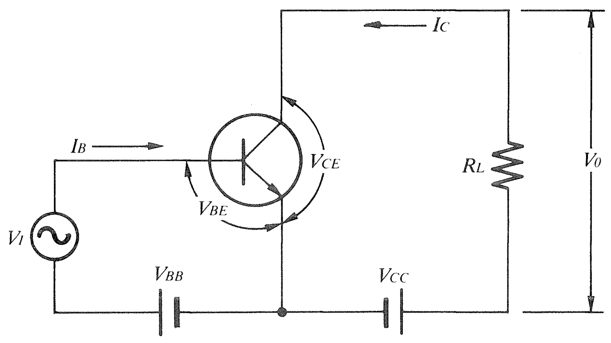
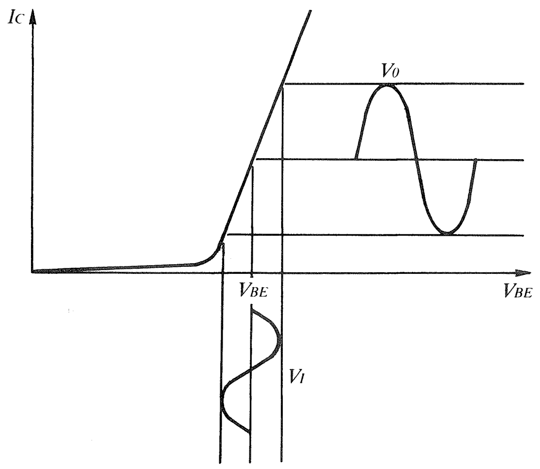
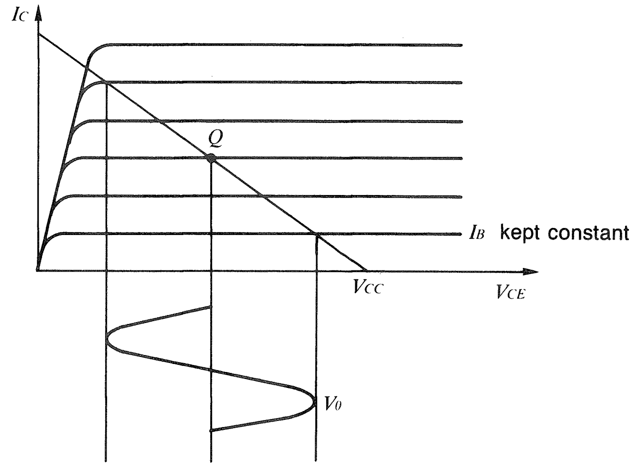


Fig. 5 Basic Bias Circuit

The above equation gives a straight line on the family of collector characteristic curves as shown in Fig. 6 (b). The operating point or quiescent point "Q" of the transistor moves on the straight line as the collector current (I_C) varies. This line is called a "load line" because the slope of the line depends on collector load resistance. To make the amplitude of the output signal maximum the operating point should be at the center of the load line by setting the V_{CE} at $V_{CC}/2$. The I_C varies greatly even when the V_{BE} varies just a little. This characteristic is employed for signal amplification.



(a) I_C-V_{BE} Characteristic



(b) I_C-V_{CE} Characteristic

Fig. 6 Static Characteristic of a Transistor

3. Bias Circuit

(a) Fixed bias circuit

This circuit supplies a base current through a fixed R_A. Its operation highly depends on temperature. When the transistor temperature increases, the operating point "Q" moves leftward making the dissipation in the transistor increase rapidly and causing distortion and thermal runaway. The dissipation increases the temperature in a vicious circle. Dissipation is the loss of electrical energy as heat. This circuit, however, is simple and consumes small power.

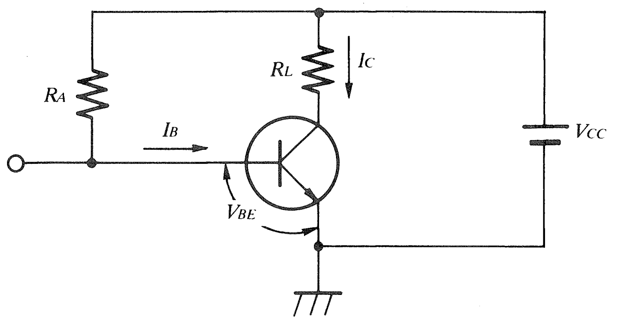


Fig. 7 Fixed Bias Circuit

(b) Self-bias (automatic) circuit

In this circuit R_L prevents I_C from increasing excessively. When I_C increases, the voltage across the R_L increases, V_{CE}, V_{CB} and I_B decrease. Then I_C decreases. Thus the operation of the amplifier is stabilized.

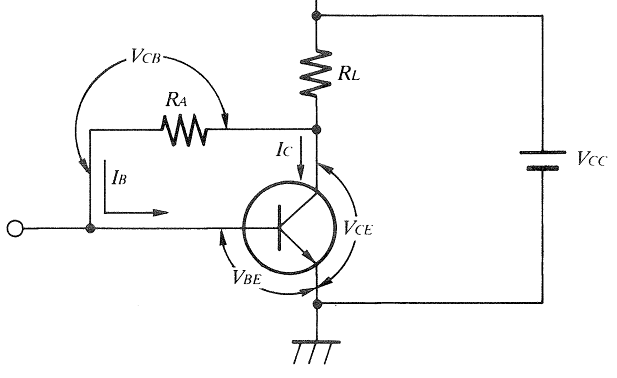


Fig. 8 Self-bias Circuit

(c) Current-feedback bias circuit

In Fig. 9 R_E stabilizes I_C. When the temperature and I_C increases, the emitter voltage increases because V_E is equal to ($R_E \times I_C$). The V_{BE} decreases, then I_C decreases.

The circuit of Fig. 10 provides a stable bias voltage and low input impedance by dividing the power supply voltage with R_A and R_B.

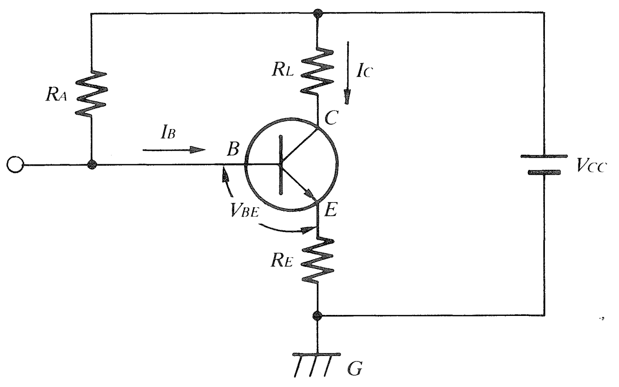


Fig. 9 Current-feedback Bias Circuit

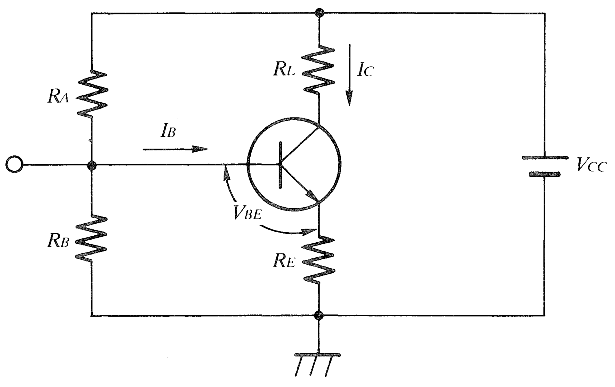


Fig. 10 Voltage-divided Current-feedback Bias Circuit

4. Basic Transistor Circuit

(a) 1-stage amplifier

Fig. 11 shows a 1-stage amplifier with a current feedback bias circuit. C1 and C2 are coupling capacitors. They pass AC signal blocking DC to avoid a circuit from affecting adjacent circuit. R_E stabilizes the I_C as discussed in (c). With the R_E the circuit composes an NFB loop for the AC input signal and decreases voltage amplification. The bypass capacitor C3 added parallelly to R_E works to minimize the NFB effect against AC and to maintain AC voltage amplification.

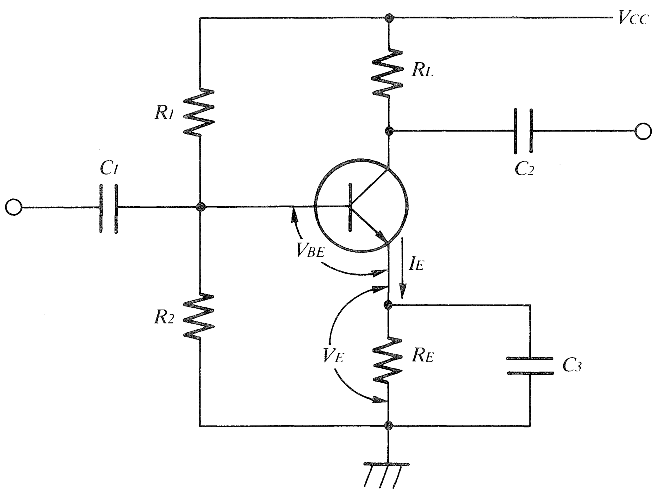


Fig. 11 1-stage Amplifier

(b) Switching circuit

In Fig. 12 a transistor works as a switch. To cut off the transistor completely $-V_B$ is applied to the base. With a pulsive signal applied to the input, V_B is negative and the transistor is off while V_I is 0V, and the base current flows and the transistor goes ON when the input level becomes V volt. This circuit is popularly used in muting circuits.

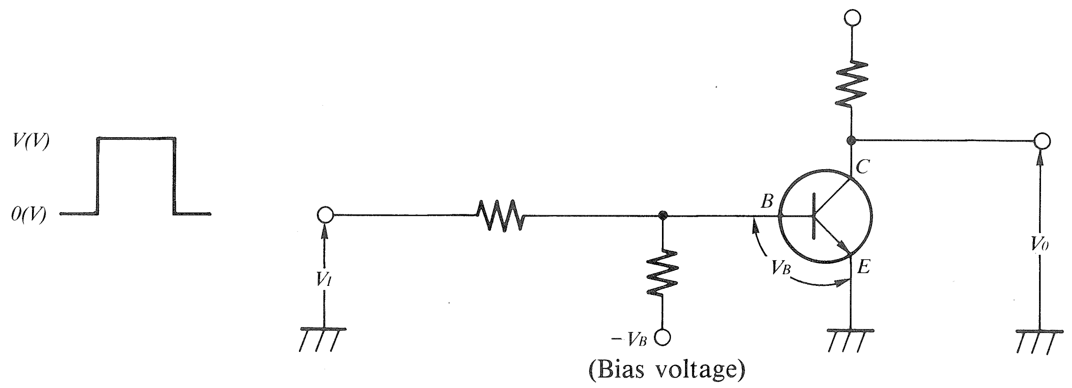


Fig. 12 Switching Circuit

(c) Ripple filter

Ripple is the fluctuating portion of the DC output voltage of electronic converters such as rectifier in power supply circuit. The frequency is equal to that of the input or internally generated switching frequency. If a power supply voltage is applied to an equipment without smoothing the ripple, it will cause humming. The R and C in the circuit of Fig. 13 (a) filter the ripple on the DC. The higher the resistance and capacitance of the R and C, the better for smoothing the ripple. However, the resistance can not be made large because a large R decreases output voltage.

In Fig. (b) a transistor solves the problem. The I_B becomes $1/(1+h_{FE})$ of the load current. This means that the R can be made large without decreasing V_O.

The relation between the currents in Fig. (b) is:

$$I_E = I_B + I_C = I_B + I_B h_{FE} = I_B (1+h_{FE})$$

$$I_B = I_E/(1+h_{FE})$$

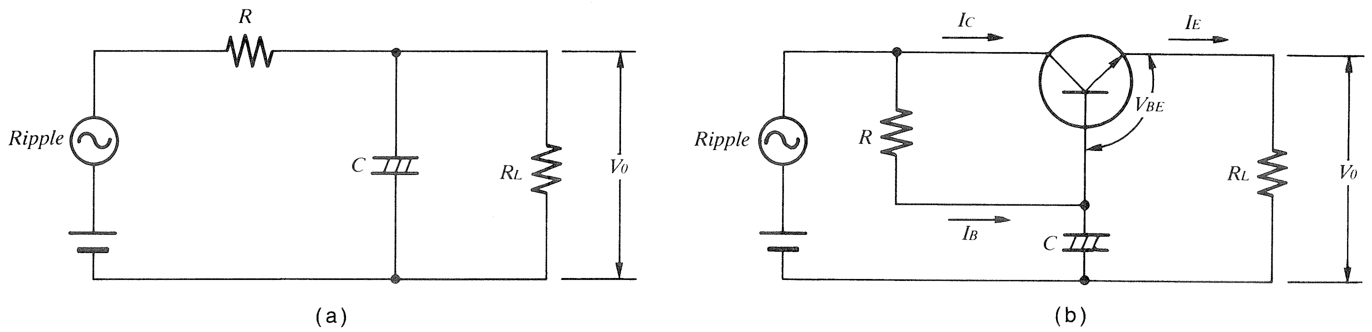


Fig. 13 Ripple Filter

(d) Voltage regulator

Voltage regulator holds its output voltage at a nearly predetermined value regardless of the changes of normal input-voltage or output load impedance. This is employed in power supply circuit to stabilize output voltage because the ability of the ripple filter is not effective enough.

Fig. 14 shows the equivalent circuit to the simplest power supply circuit with a power source and load connected. Here the output voltage (V_O) can be obtained by:

$V_O = V_I - I_O R_S$
V_I: Input voltage
I_O: Load current
R_L: Load resistance
R_S: Internal resistance of power source

I_O varies depending on the input voltage and load impedance and affects the output voltage.

The circuit of Fig. 15 maintains the V_O by varying I_O with the help of a stabilizing transistor. When V_O decreases, V_{BE} decrease because V_S is constant, then I_O decreases and thus V_O decreases to a predetermined level.

Fig. 16 shows a typical voltage regulator circuit. The battery (V_S) has been superseded by a zener diode which prevents the base voltage from increasing above 0.7V. When V_I fluctuates, the ripple component in the input current flows in the zener diode and does not affect the output. This circuit is a common-collector circuit. The output impedance of this circuit is low and the effect of the load current fluctuation to the output voltage is kept low. The output voltage is:

$V_O = V_Z - V_{BE}$

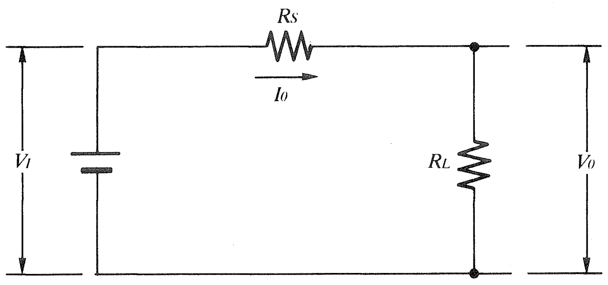


Fig. 14 Equivalent Circuit to the Power Supply Circuit

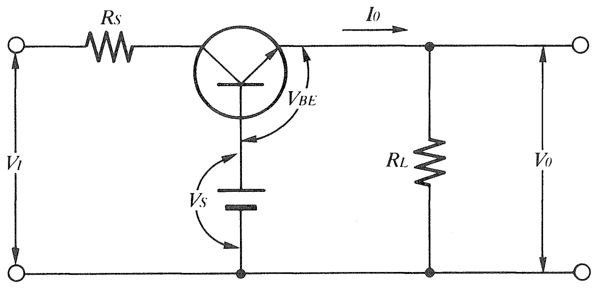


Fig. 15 Voltage Regulator With a Battery

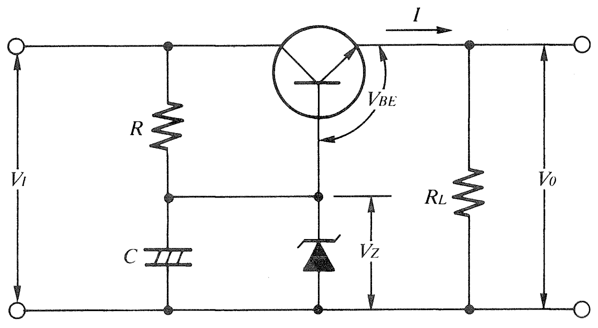


Fig. 16 Voltage Regulator With a Zener Diode

By H. Kishida

Measuring Instrument

Wow-flutter Meter

We sometimes experience that the sound on a tape or disc trembles or varies like that of the old movies. It is a kind of modulated distortion or frequency deviation produced by irregular motion of the driving mechanism of a turntable or tape transport during recording or reproduction. This is called "wow and flutter". The problem has almost been solved by a phase-locked loop servo system which converts speed error signal to a corrective voltage. Here we will discuss Wow Flutter Meter basing on MEGURO MK-668E.

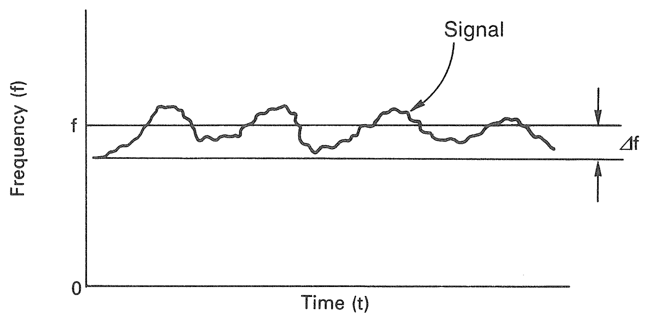
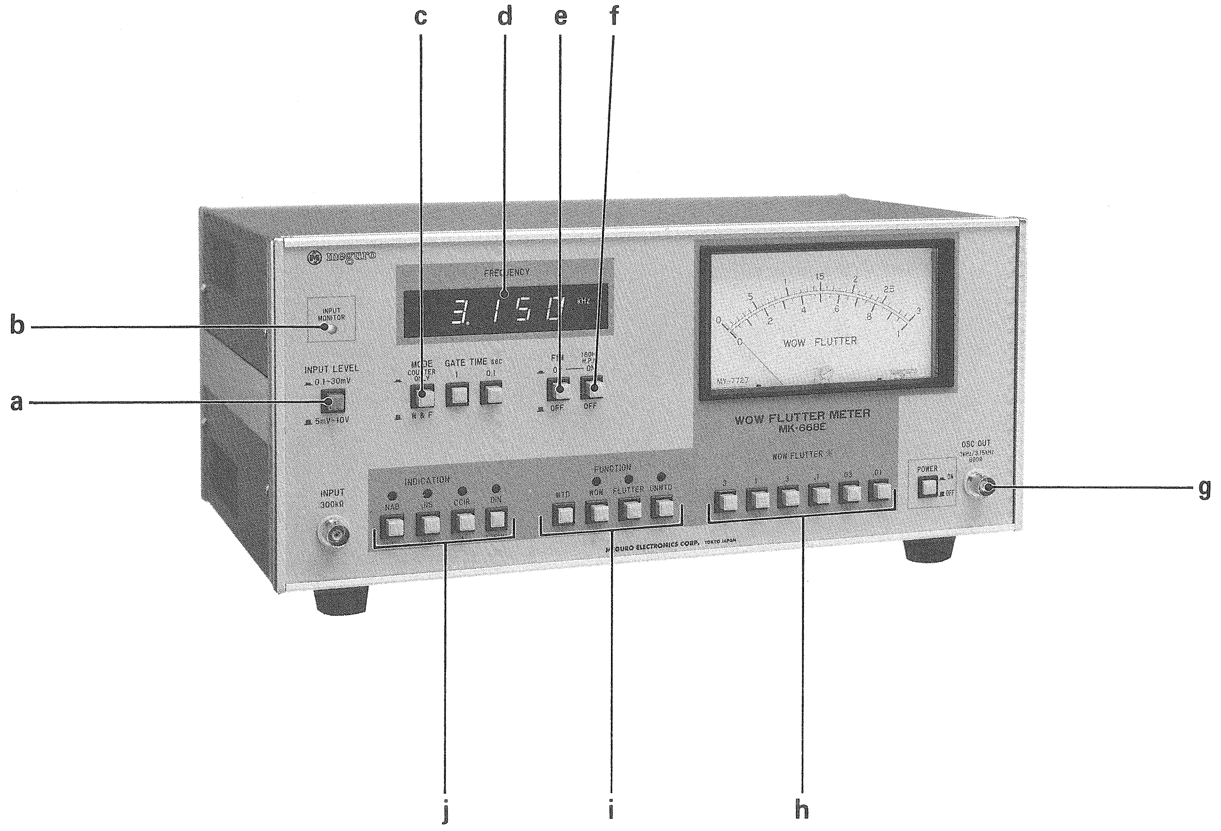


Fig. 1 Sound Pitch Variation



1. Definition

IEC-1969 (International Electro-technical Commission) defines the wow and flutter as follows:

Flutter: Unwanted frequency modulation or deviations of signals of 10Hz or above produced by irregular motion of a recording medium during recording or reproduction

Wow: Unwanted frequency modulation or deviations of signals in the range of 0.1Hz – 10Hz produced by irregular motion of a recording medium during recording or reproduction

Drift: Slow and random speed variation of a recording medium close to 0Hz

Wow and flutter is expressed by the percentage of the frequency deviation to the input frequency as follows:

$$\text{Wow and flutter} = \frac{\Delta f}{f} \times 100 \quad [\%]$$

f : Input frequency
Δf: Frequency deviation

The above relation can be converted to voltage relation by a frequency discriminator:

$$\text{Wow-and-flutter: } \frac{\Delta V}{V} \times 100 \quad [\%]$$

V : F-V converted DC voltage
ΔV: F-V converted AC voltage of wow and flutter component

The sensitivity of our ears differs individually and depends on the conditions such as the materials, size and shape of listening room (live or dead), sound quality and level. Fig. 2 shows the percentage of the people sensed the wow and flutter under various conditions. (a) indicates that 50% of the people tested sensed the wow of 0.2% on the 3Hz signal. It is strange that some people feel wow and flutter even when it is 0%.

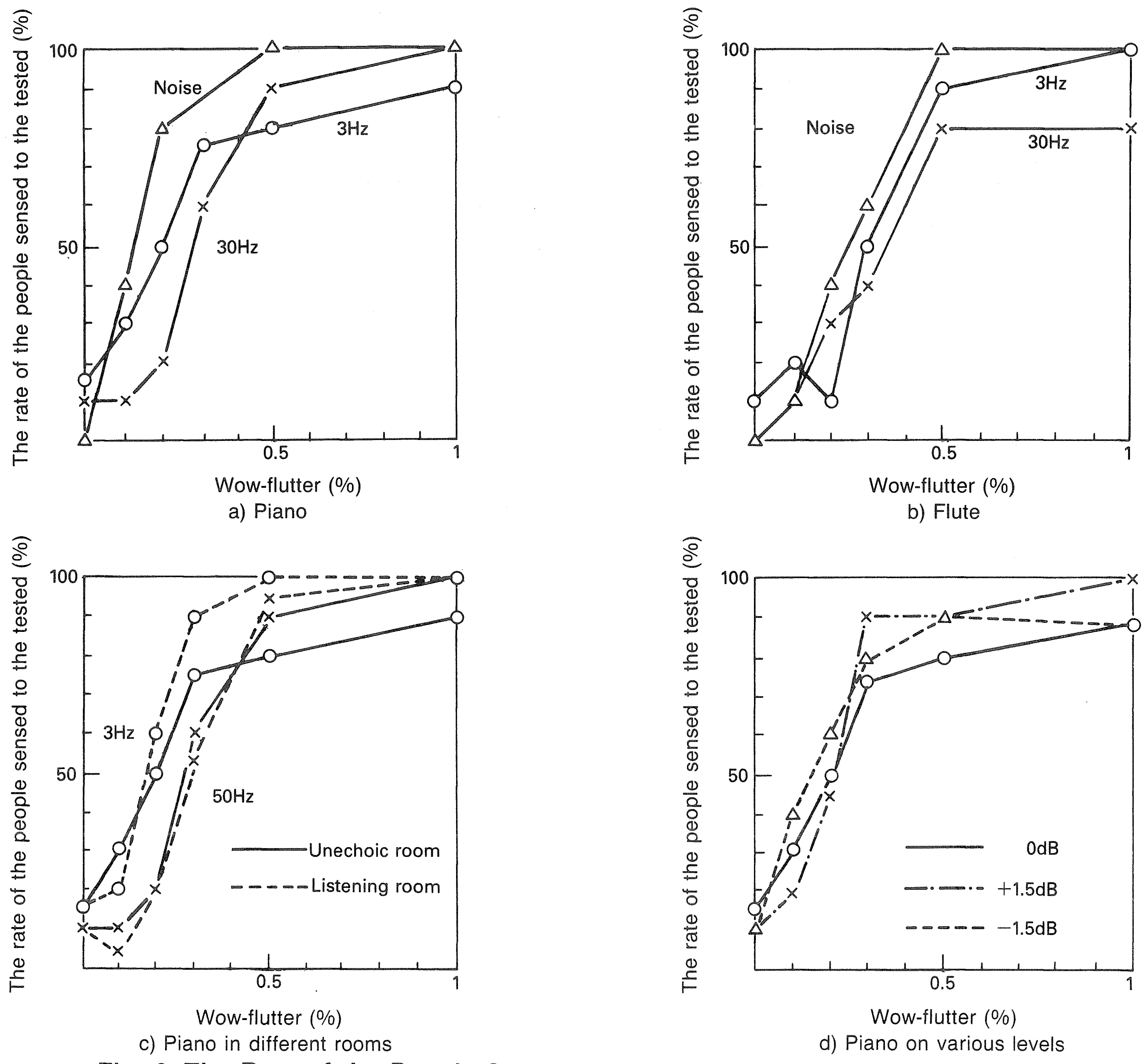


Fig. 2 The Rate of the People Sensed the Wow and Flutter to the Tested

2. Panel Descriptions

a. INPUT LEVEL Input level range is selectable between 0.1V-30mV and 5mV-10V.
b. INPUT MONITOR When the input level is high enough, the LED lights.
c. MODE Selects wow and flutter meter or frequency counter.
d. FREQUENCY Input signal frequency counter. This can count from 0.001kHz to 9.999kHz or from 0.01kHz to 99.99kHz with the GATE TIME set at 1 sec or 0.1 sec respectively and is useful as an independent frequency counter.
e. FIM Frequency intermodulation can be measured with this button depressed.
f. 160Hz HPF Depress this to eliminate wow and flutter component when measuring FIM.
g. OSC OUT Standard signal is available here. 3kHz signal for NAB, JIS and CCIR and 3.15kHz for DIN.
h. WOW FLUTTER % Six ranges are selectable.
i. FUNCTION
WTD To weight* input signal.

**Weighting: The artificial adjustment of measurements in frequency characteristic to meet the sensitivity of the human ears. In 1956 CCIR adopted the DIN's weighting standard as an international standard. Since then many nations and associations such as IEC, JIS, NAB have adopted the weighting curve as their standard. Fig. 3 shows the curve.*

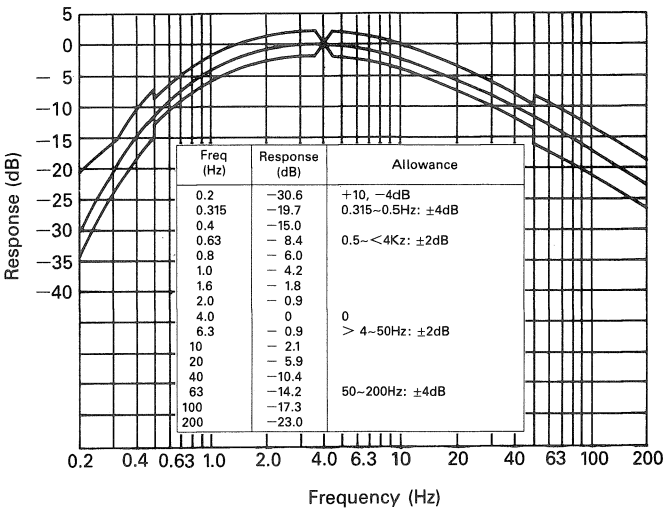
WOW	To measure wow only (0.5 — 6Hz)
FLUTTER	To measure flutter only (6 — 200Hz)
UNWTD	To measure wow and flutter without weighting

j. INDICATION The reading depends on the standard selected from NAB, JIS, CCIR and DIN.

CCIR:	International Radio Consultative Committee
DIN:	Deutche Industrie Normen
IEC:	Internation Electrotechnical Commission
JIS:	Japanese Industrial Standard
NAB:	National Association of Broadcasters

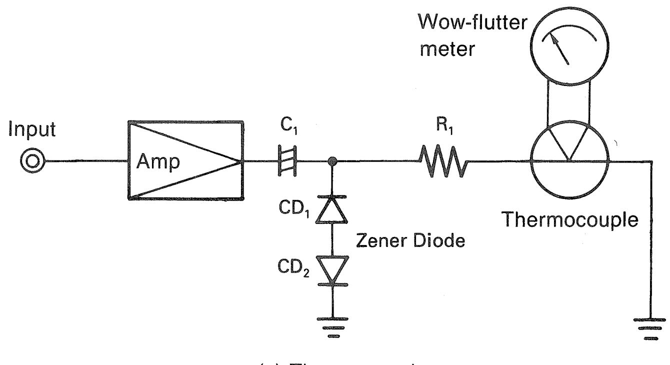
There are three main ways of indication; rms rectification — rms (root-means-square) indication by the JIS, mean-value rectification — rms indication by the NAB, and peak-value rectification — peak-to-peak (p-p) indication by the DIN. The DIN is widely adopted by many European CCIR member countries. Japan has recently adopted the DIN. They are different each other. Let's see the typical circuits.

RMS indication: Fig. 4 shows popular rectifiers which operate with a time constant made by C_1 and R_1. Unlike a thermocouple, RC rms circuit has many advantages such as linear scale and wide selection of meters in dynamic characteristic. In the characteristic, the JIS requires lenient indication of 100±5% with the input pulses of 5-second width.

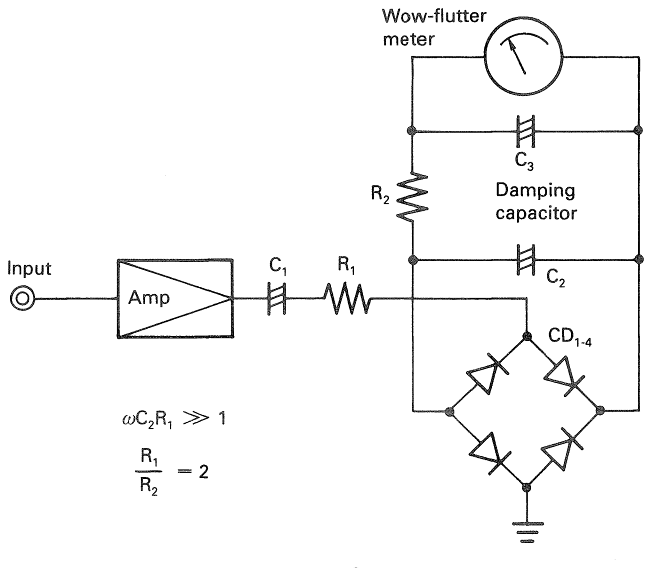


Freq (Hz)	Response (dB)	Allowance
0.2	−30.6	+10, −4dB
0.315	−19.7	0.315~0.5Hz: ±4dB
0.4	−15.0	
0.63	− 8.4	0.5~<4Kz: ±2dB
0.8	− 6.0	
1.0	− 4.2	
1.6	− 1.8	
2.0	− 0.9	
4.0	0	0
6.3	− 0.9	> 4~50Hz: ±2dB
10	− 2.1	
20	− 5.9	
40	−10.4	
63	−14.2	50~200Hz: ±4dB
100	−17.3	
200	−23.0	

Fig. 3 Weighting Characteristic



(a) Thermocouple type



(b) RC type

Fig.4 RMS-indication Circuit

Mean-value rectification: The full-wave rectifier shown in Fig. 5 gives mean value. The NAB requires to employ a VU meter. Reading the VU meter is very hard because its pointer moves very quickly.

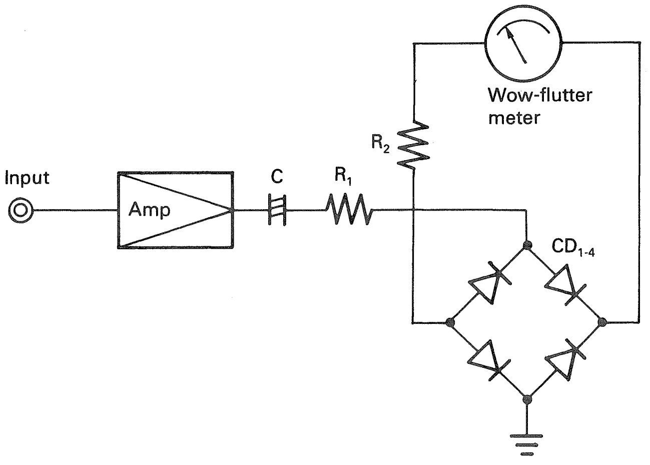


Fig. 5 Mean-value Rectification Circuit

p-p-value indication: The circuit of Fig. 6 employs a voltage doubler to indicate p-p value. The DIN strictly prescribes the dynamic characteristic of peak indication circuit. Fig. 7 shows its characteristics. The pointer should swing up to the full scale in 100ms when a large pulse is applied and return to the 40% point in 100ms after the pulse disappears.

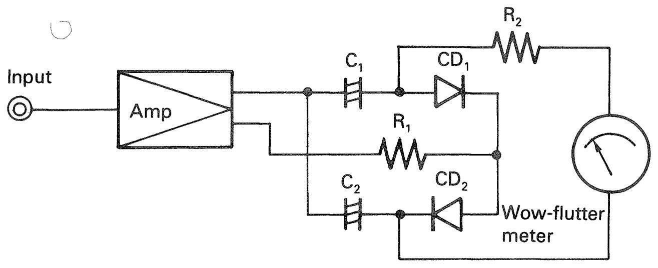
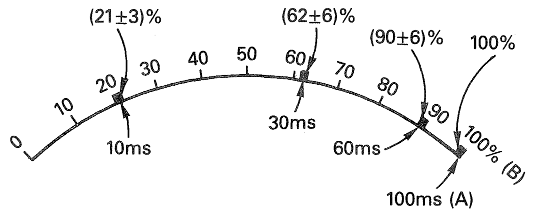
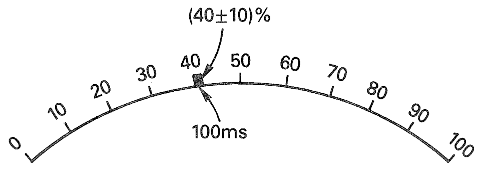


Fig. 6 p-p-value Indication Circuit



(a) Charging Time



(b) Discharging Time

Fig. 7 Dynamic Characteristics of p-p Indication

k. TO SCOPE (rear panel)
The waveform of the wow and flutter signal is observable on an oscilloscope taking its signal from this terminal.

l. TO RECORDER
Frequency demodulated signal is available here.

3. Principle

A wow meter has the same frequency demodulator as those of FM tuners because the wow and flutter are FM signals. A 3kHz standard signal recorded on a test disc or tape is reproduced. The 3kHz signal will be FMed and AMed in the turntable or deck to be measured. Its output is applied to the wow meter.

In the meter the signal level is adjusted and its amplitude is limited to eliminate the effect of AM component. Wow-and-flutter component is taken out of the limited signal by a 3kHz frequency discriminator. The wow-and-flutter signal is attenuated to meet the selected meter range. It is amplified and is weighted in accordance with the standard selected from NAB, JIS, CCIR and DIN. Wow and flutter can be separately taken out by a 6Hz LPF or HPF. Fig. 9 shows their characteristics. The signal is rectified and indicated as specified by the respective standard selected. The current models can measure tape speed and motor's revolution with a built-in frequency counter.

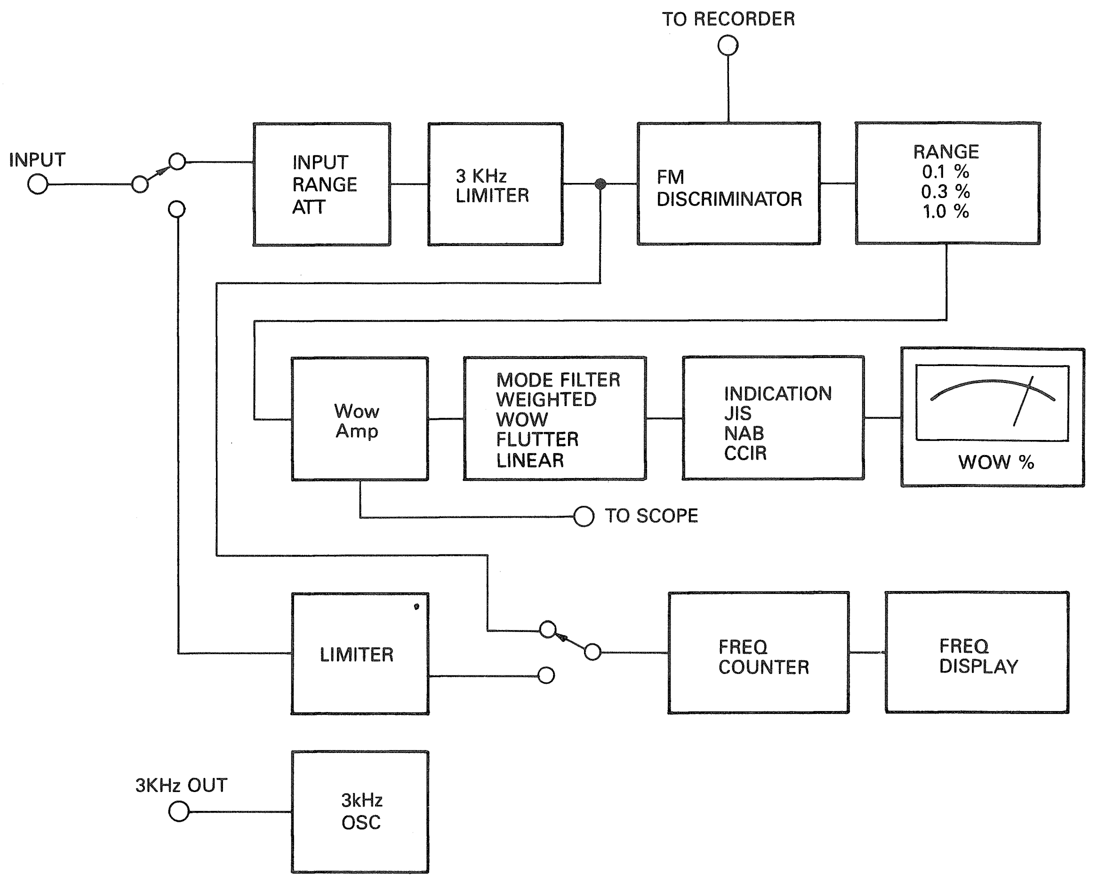


Fig. 8 Block Diagram of Wow Meter

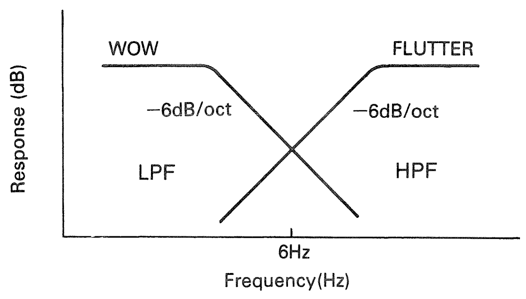


Fig. 9 Characteristics of LPF and HPF

4. Measuring a Tapedeck

Measurement with a test tape by the JIS (different recorder/player):

The JIS measurement uses a test tape with a 3kHz signal recorded by a standard recorder. In this way the tape is played on a tape deck other than the one recorded. A test tape is played on the unit to be tested, the output of the unit is weighted by the JIS, measured and displayed in rms percentage on the meter as shown in Fig. 10. The test tape should be on a reel of the largest type installable and the test should be made at the beginning, midway and the end for at least 10 seconds at each point, and the largest value among the three should be taken. Many deck manufacturers follow this way.

Simultaneous recording/playback:

In case the total value throughout recording and reproduction is required, a 3kHz signal is recorded on a tape, it is rewound and then reproduced on the same unit to be tested. The measurement is repeated several times, and the worst value except those measured when starting is taken. In this way accurate measurement will be difficult because particular frequency components determined by head gap and the wave length of flutter signal will be emphasized or suppressed. ↗

Nonsimultaneous recording/playback:

In this way the measured value may vary from time to time due to the signal phase difference between recording and reproduction. DIN employs this method. The meter pointer will not become stable because the wow-flutter signal contains various frequency components.

Electro-mechanic Testing Association stipulates it as the maximum reading within 20 seconds.

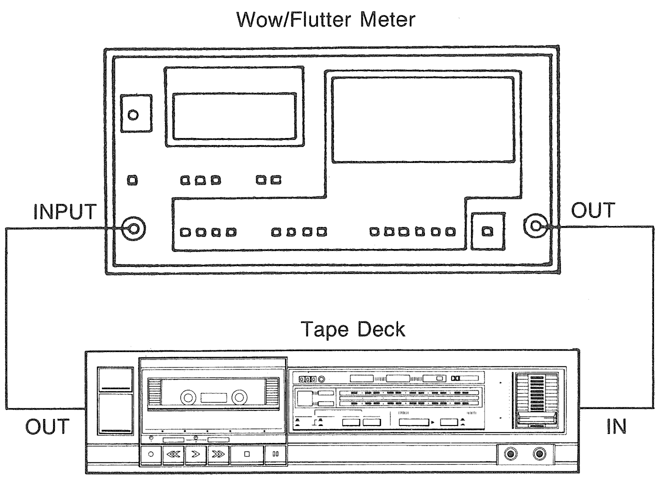


Fig. 10 Wiring Diagram for Measuring a Tape Deck

5. Simplified Calibration of Wow Meter With a Tape

Rough calibration of the meter can be made with a test disc or tape recorded with a 3kHz signal added with the standard 3% wow-and-flutter signal. They are available on the market. A test tape can be made by recording the signal from the disc to a tape. The turntable and tape deck should be high in performance. The distortion of the deck should be less than 0.1%. The distortion of less than 0.1% added by the deck is negligible.

Play the disc or tape on the high-performance deck, apply the 3kHz signal that includes the wow-flutter signal to the INPUT of the wow meter to be calibrated and turn the screws in the holes above the INDICATION and FUNCTION knobs as indicated in the instruction manual attached to the meter. Remove the screwdriver when the meter indicates 3% distortion.

By T. Kubota

One Point Service Technique

Diagnosing Optical Pickups

High precision is required in adjusting the Pickup of LD and CD players after replacing it or Spindle-motor because laser beam should be converged to $\phi 1.5\mu m$ to read pits of $\phi 0.5\mu m$. Here we will discuss the ways to find which component is faulty, Pickup or mechanical component, and to see if they are properly adjusted.

1. Optical Pickup

The trouble of Pickup is caused by beam path deviation which prevents the beam reflected on the disc from reaching the center of the Photo Detector or by faulty components inside the Pickup. Faulty Pickups should be replaced because replacing the components inside the Pickup is possible only in our factory. Care should be taken in judging a Pickup to be faulty because it costs very much.

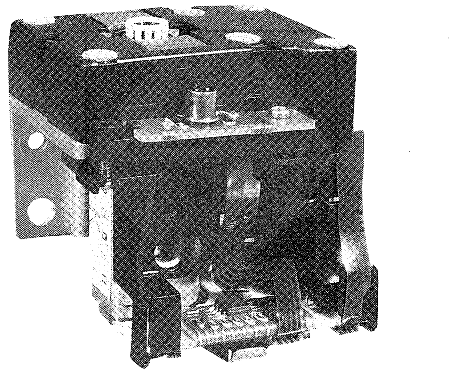


Photo 1 Optical Pickup

2. Pickup Problems

The symptoms of the disorder of Pickup are:

- **a. Laser diode does not turn on.**
- **b. Lens does not move up and down when starting.**
- **c. FOCUS Servo does not lock.**
- **d. SPINDLE Servo does not lock.**
- **e. Beam skips tracks.**
- **f. SEARCH is inoperative.**
- **g. Picture is noisy.**

Let us troubleshoot them.

a. Laser diode does not turn on.

Check point: Are APC (Automatic Power Control) current and LD ON signal applied to Pickup?

Yes: Replace Pickup because the trouble is in Laser Diode or APCB in Pickup.

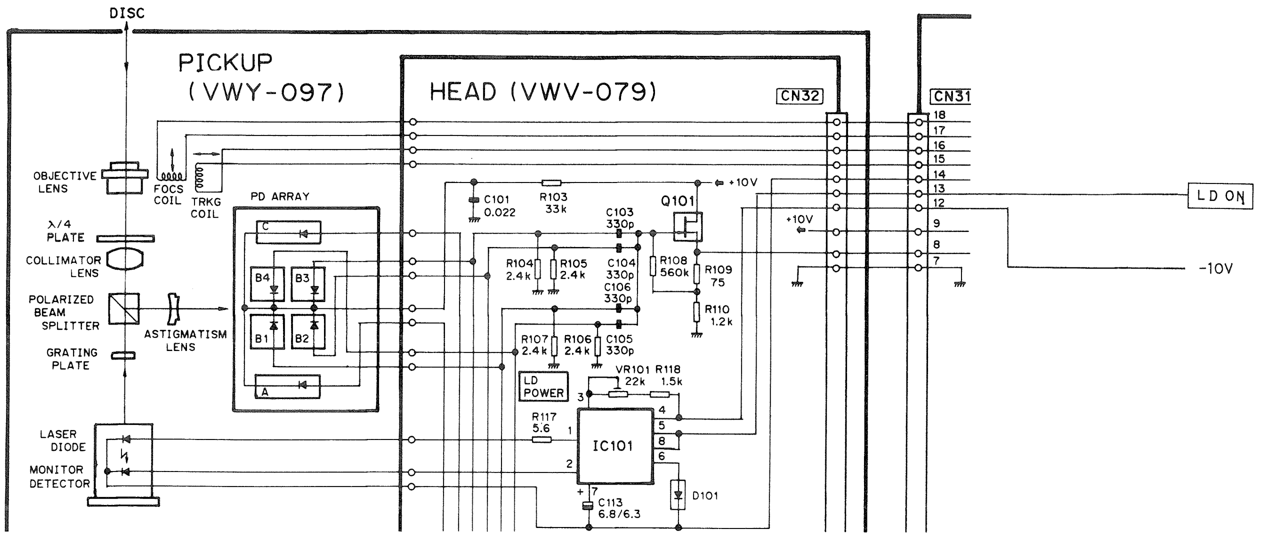


Fig. 1 Circuit of Pickup

No: Faulty power supply block. The following conditions should be satisfied to apply LD ON signal to Pickup. Verify them.

LD-1100 series:	INTER-LOCK Switch: ON
	LID-OPEN Sw: ON
LD-700, CLD-900, etc.:	Disc Sensor, which also senses 8″/12″ disc size, operates.
LD-707, CLD-909, etc.:	Disc Sensor, which also senses TILT, operates. Door Sw is closed.
P-D70 series	INTER-LOCK Sw: ON.
PD-5010, 5030 series	CLAMP Sw: ON.

b. Objective-lens does not move up and down.

Check point: Is RAMP-generator working and a Drive-current applied to Focus-coil?

Yes: Shorted or opened Focus-coil, or sticking Lens.

No: Faulty RAMP-generator circuit

Note: Ramp Generator

Focus-Servo circuit works only when it receives Focus-Error signal from Pickup. Ramp-Generator generates a voltage to move Objective-lens up and down into the height range where focus lock is possible. This is activated when LD ON signal is applied, stops when Focus-Error signal is applied and then activates Focus Servo. If Focus-Error signal is unavailable, this circuit will stop generating Ramp signal in a few seconds and give up focusing. The player goes into STANDBY mode or EJECT mode.

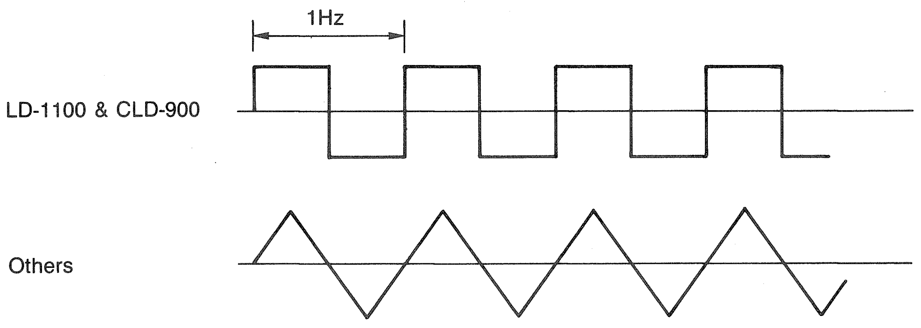


Fig. 2 Focus-drive Signal

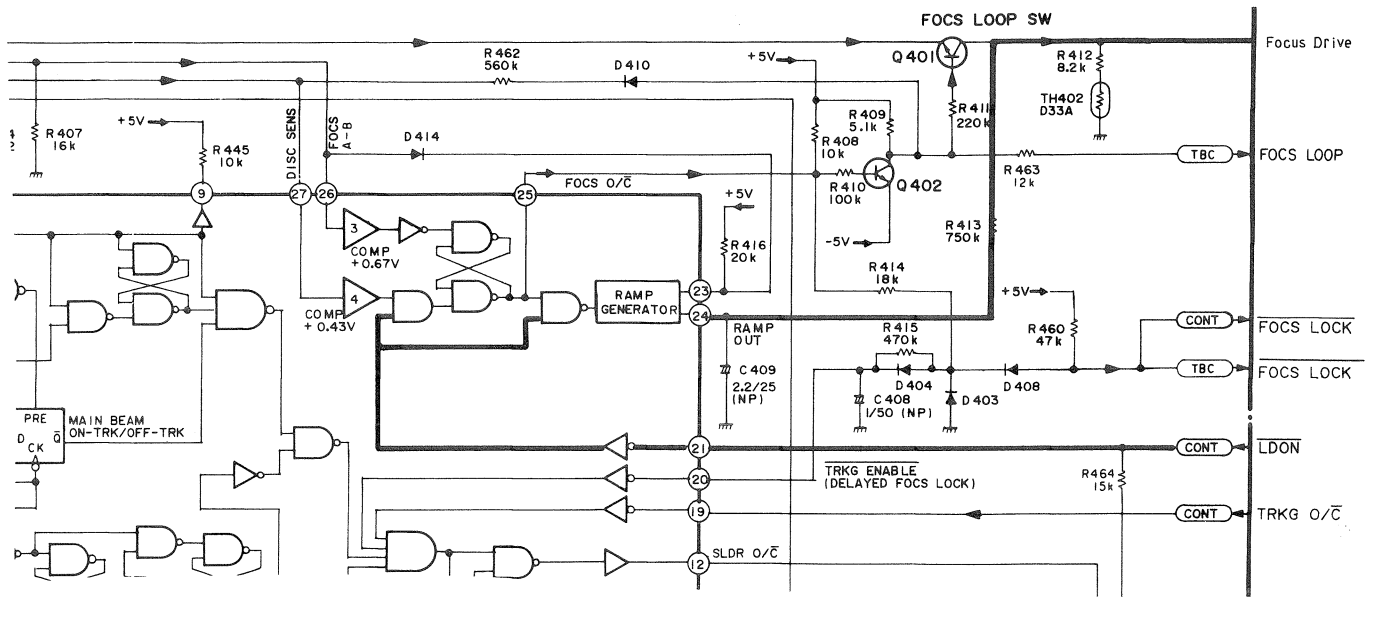


Fig. 3 The Route of $\overline{\text{LDON}}$ Signal of LD-700

c. Lens moves up and down, but does not focus-lock.

Check point: Observe the levels of FOCUS (A − B) and TRKG (A + B) signals on an oscilloscope while Lens is moving up and down.

If the Pickup is normal, an S-shaped wave shown in Fig. 4 (a) can be observed. The wave should be symmetrical and large in amplitude. If the wave is unsymmetrical and less than 2Vp-p, the beam axis must be deviated. → Replace it.

With a normal Pickup, the focus locks only when the amplitude of TRKG (A + B) signal is more than 1.5 Vp-p and the voltage shifts from negative to positive.

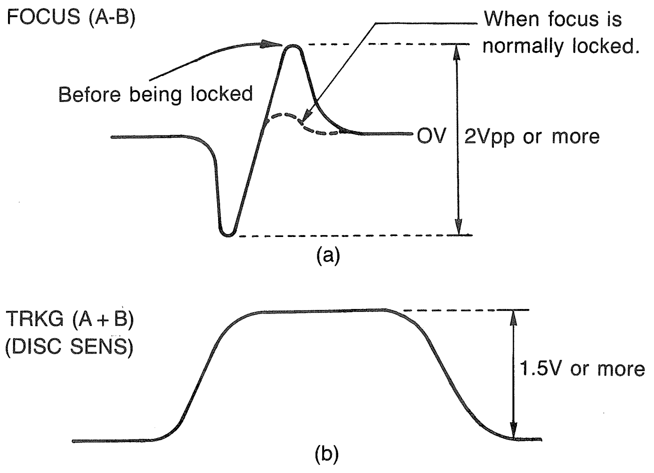


Fig. 4 Focus and Tracking Signals

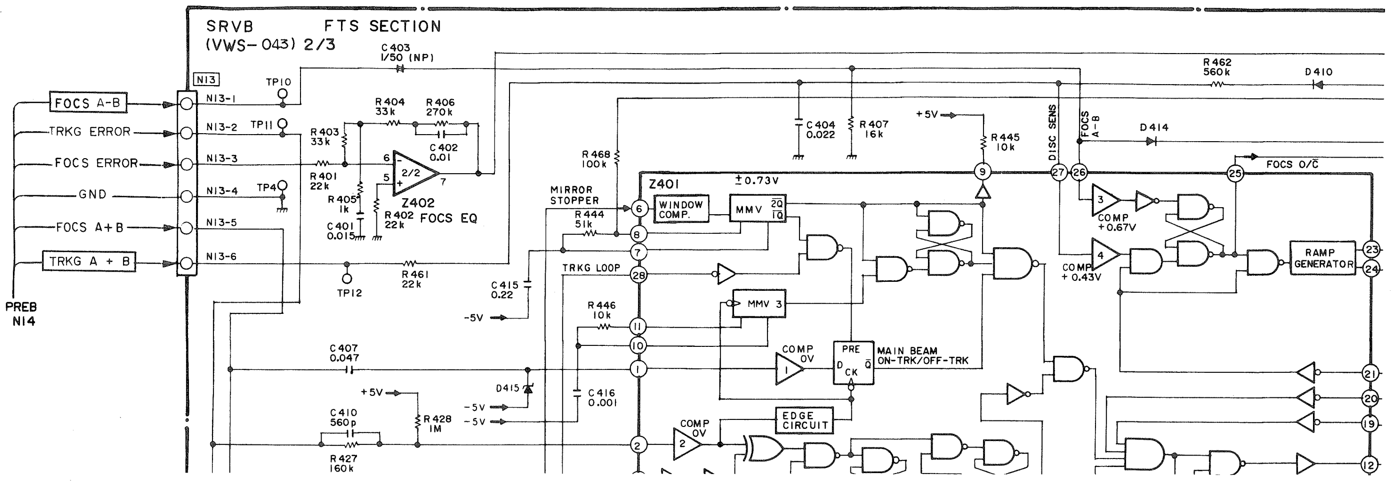


Fig. 5 The Signal Route of FOCUS A-B and TRKG A + B Signals (LD-700)

d. — g) Refer to the following.

3. Checking Optical Axis (Beam path)

Requisites to good Pickup

a. Pickup should be capable of producing a tracking error signal with a clear and flat envelope of high amplitude when the tracking loop is opened. (More than 7Vp-p at the Frame #20000 of a test disc of F1 — F5 played on CLD-900)

b. The reflected beam should shoot Photo-detector in the center.

c. The beam should converge to $\phi 1.5\mu m$ on the disc when Focus Servo locks.

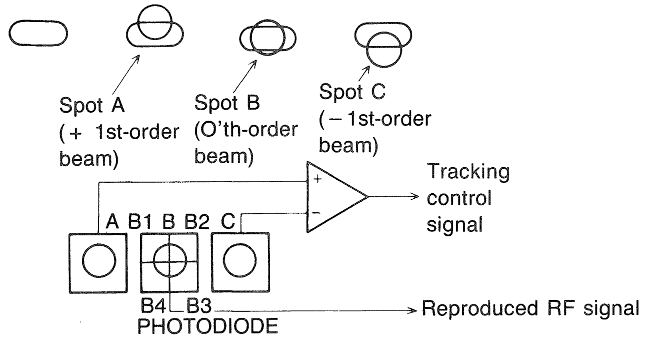
d. The beam passing Objective-lens should be round.

If all of the above are satisfied and the trouble still exists, the cause of the trouble must be in other block than Pickup.

Checking methods of Pickup are as follows.

3-1 Grating adjustment

The Grating splits one beam coming from Laser Diode into three; the 0'th-order beam is for reading RF signal on a track and for generating Focus-error signal, and the ± 1st-order beams are for tracking both sides of the track to keep the center beam trace the track center. Grating adjustment lets the three beams shoot a track as shown in Fig. 6. If this adjustment is improper, Tracking Servo will become impossible and the beam will skip tracks or the player will not play.



Checking Grating Adjustment

a. Apply TRKG (A − B) and TRKG (A + B) to X and Y channels of the oscilloscope respectively. Select X-Y mode.

b. Play a test disc. Open TRKG SERVO at the frame shown below.

Model	Frame Number
LD-1100 series	18,000
LD-7000 series	15,000
CD	Midway

c. Observe Lissajous figures. The figure will be a horizontal line when the adjustment is optimum. The figure also becomes a horizontal line even when the two tracking beams (± 1st-order Beams) are on the side tracks. Usually they do not deviate so much.

d. Referring to the service manual, adjust it finely if the figure is thick or round in the vertical direction. If it becomes round on inner or outer tracks, adjust the motor position. This will be discussed later.

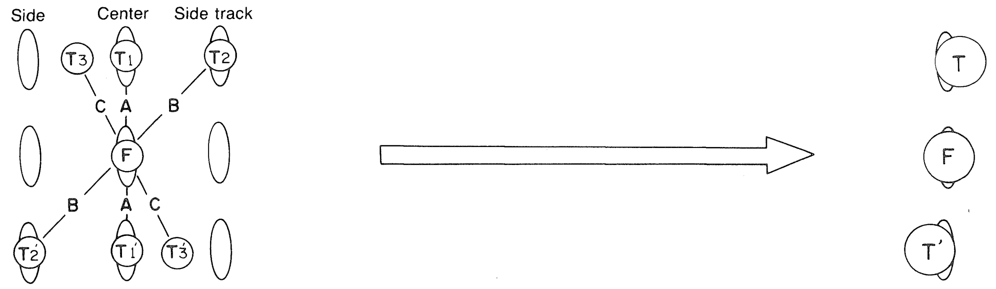


Fig. 6 Three Beam Positions

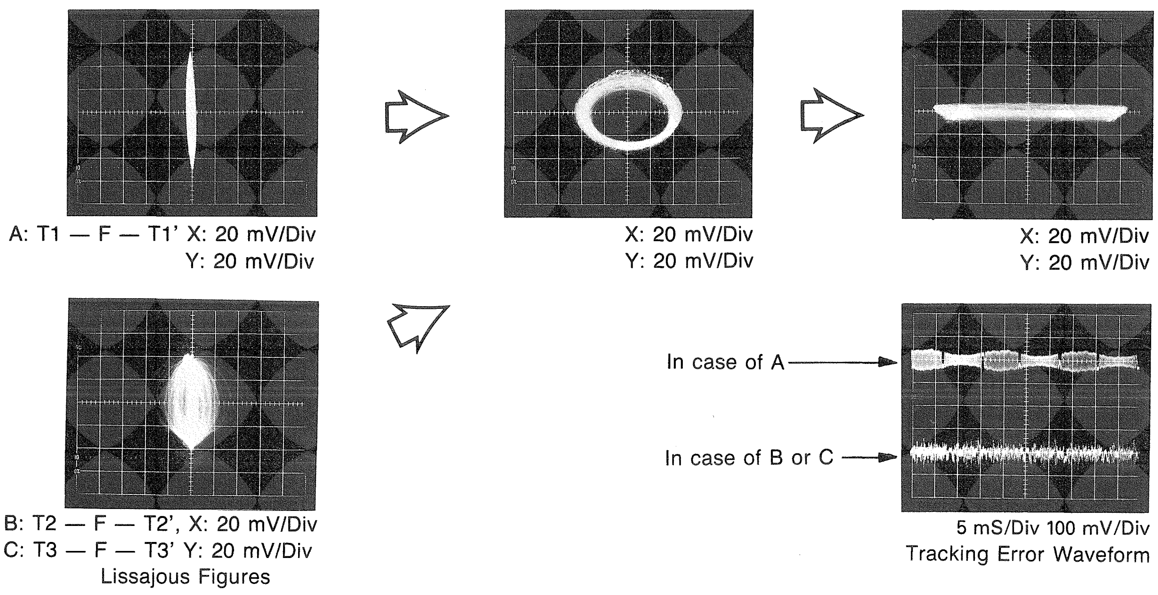


Photo 2 Lissajous Figures Depend on Beam Positions

3-2 Checking Beam Path (LD Player only)

a. PD Chopper

This is for verifying on an oscilloscope that the 0th-order beam is shooting the 4-segment photo detector in the center by comparing the levels of the signals taken from the four segments.

Note: PD Chopper (GGV-083)

PD chopper is the jig to adjust the photo detector's position properly. This jig was designed for the LD-1100 series models originally, however, it can be used for adjusting LD-700 and CLD-900.

(1) Connect PD Chopper, oscilloscope and Player as illustrated below.

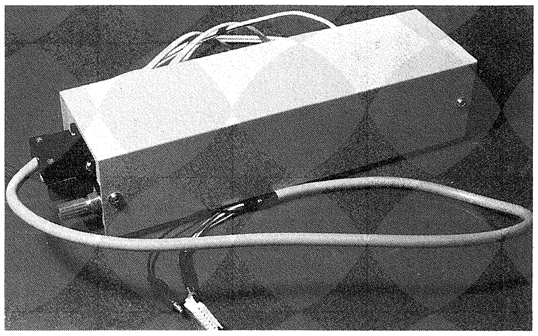


Photo 3 PD Chopper

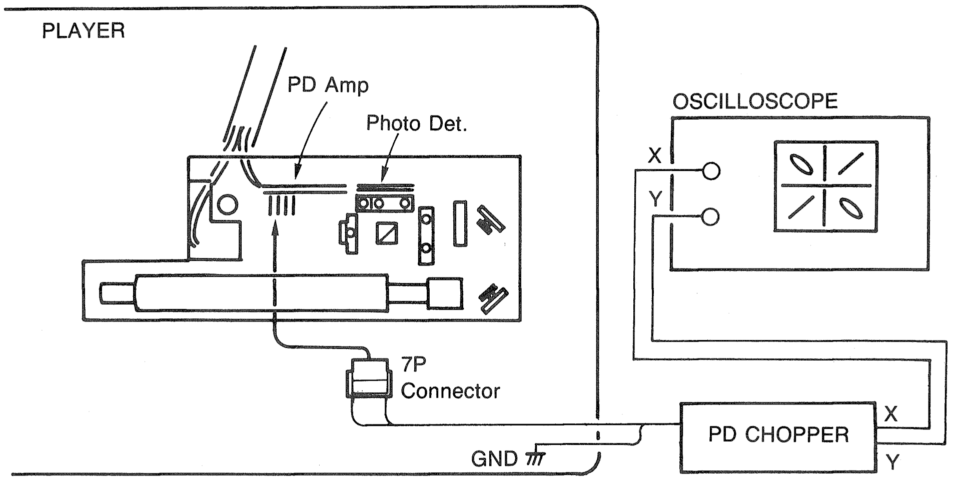


Fig. 7 Connecting Diagram of LD-1100 or LD-660

The PD chopper requires an adaptor to be connected when adjusting other players than LD-1100 series. It is connected to the Photo-Detector-output terminals B1 — B4.

If you have no adaptor, connect as below. The B1 — B4 of other models can be found easily on the circuit diagram.

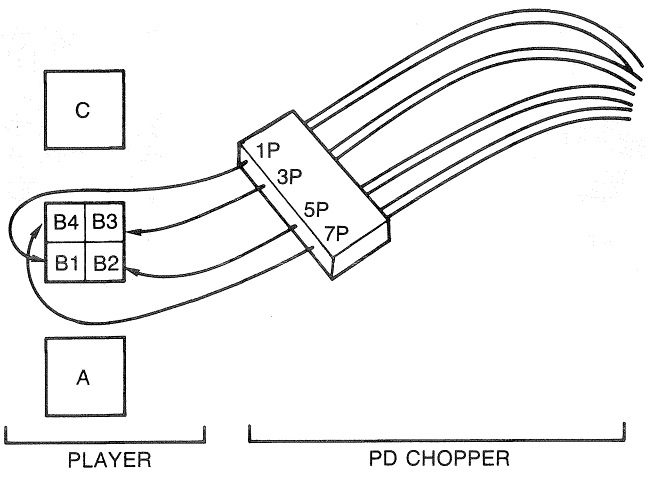


Fig. 8 Connecting Pins with Photo Segments

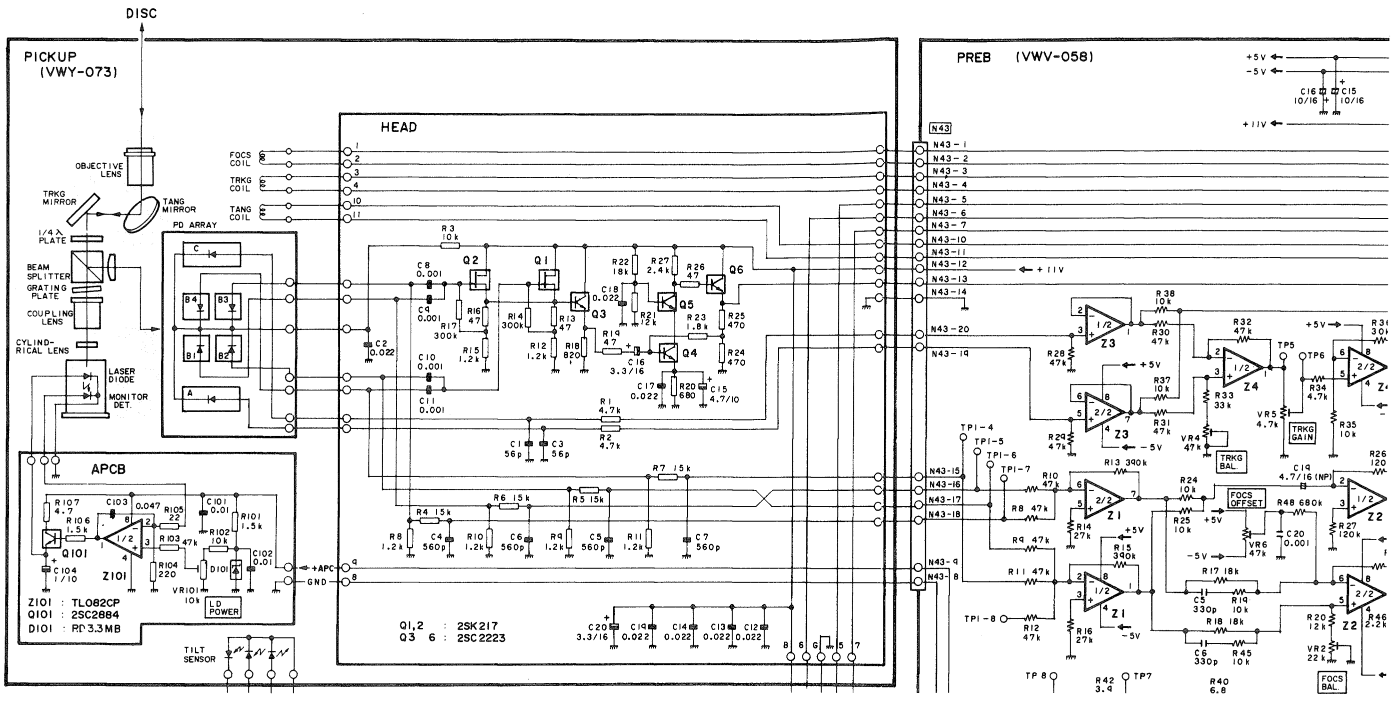


Fig. 9 Pickup Circuit of LD-700

Chopper's Terminal	PREB
Pin 1	TP1-5
(B1)	
'' 3	TP1-7
(B3)	
'' 5	TP1-6
(B2)	
'' 7	TP1-4
(B4)	

The Jig number of the adaptor for LD-700 and CLD-900 is GGF-057.

(2) Turn on the power of Player, PD chopper and oscilloscope.

(3) Select X-Y and DC input modes of oscilloscope.

(4) Move the beam spot to the center of the screen turning the POSITIONS.

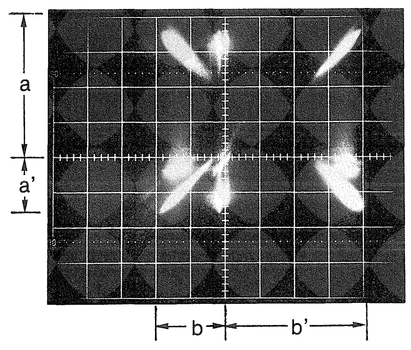
(5) Play a test disc. Open the Tracking Servo in the midway on the disc.

(6) Observe waveform.

(7) If the beam path is deviated,
LD-1100 & -660: Adjust them with PD Adjusting Screw.
LD-700& CLD-900: The Pickup can be adjusted separately with the following jigs.

Mirror Jig	GGF-056
Adaptor Cord	GGF-057
PD Chopper Jig	GGV-083
1.27mm Hex Wrench	GGF-060

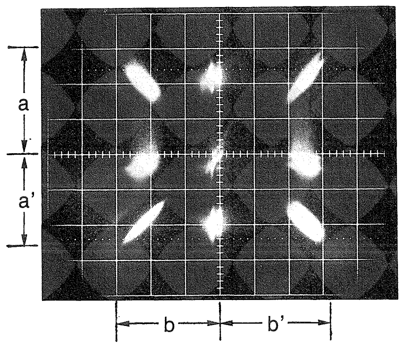
Deviated Pickups mounted on the models other than the above should be replaced.



No good

If a:a' or b:b' can not be made smaller than 3:2 or 2:3, the Pickup may be faulty.

(a)



All right

If the Lissajous figures are symmetrical across the center point, the Pickup is good. a = a', b = b'.

(b)

Photo 4

b. Mirror Biasing

Beam path can be checked on the players which have TRKG and TANG mirrors such as LD-1100, LD-700 and CLD-900 by applying a bias current to the mirrors and making them oscillate.

The waveform of the Tracking Error signal tells us whether the beam path is shooting the mirrors' centers. It also tells us if the Mirror Actuator is worn out.

Refer to individual service manual for details of this method. Before using this method verify the Grating adjustment. Readjust it if it is improper. Correct judgement of the beam path by oscillating TANG mirror will be difficult if the Grating has been misadjusted.

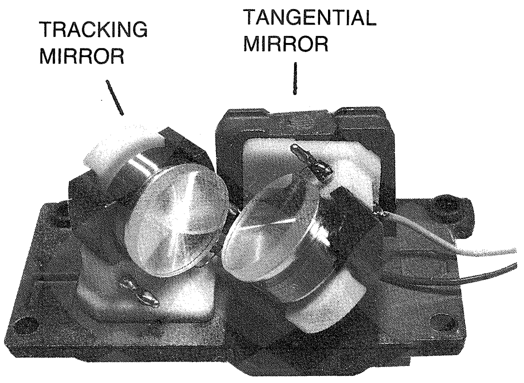


Photo 5 Tracking Mirror and Tangential Mirror

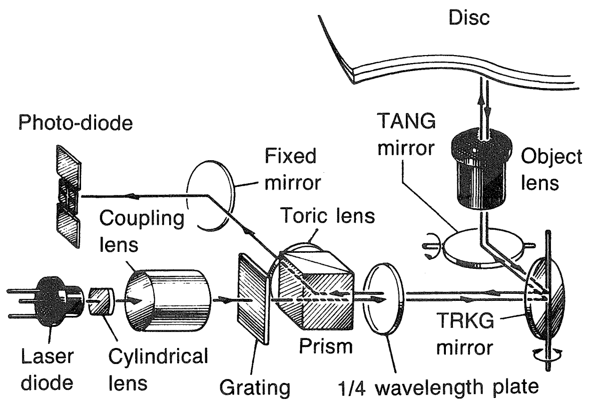


Fig. 10 Beam Path

c. Checking LD-700

PICKUP OPTICAL AXIS CHECK

Always perform this procedure after replacing the pickup and when it is suspected that the pickup has been deviated.

- Play a disc at around the track number 15,000.
- Open the TRKG loop. (Connect SRVB, Z401, PM4001 pins 20 and 22 with shorting clips.)
- Open the TANG loop. (Connect SRVB TP7 to ground.)

- If Vbm is within the range of ±2.4V: $E_p > 0.63E_0$ and $E_n > 0.63E_0$
- If Vbm is outside the range of ±2.4V: $E_p > 0.70E_0$ and $E_n > 0.70\ E_0$
- If the above conditions are not satisfied replace the pickup.

VERIFYING OPTICAL AXIS IN TRACKING DIRECTION

- Connect the bias-voltage output terminal of the optical-axis-checking-jig (the current setting resistor should be set to 200 ohms) to TP1 (TRKG RTN) of SRVB.
- Measure the TRKG error level at TP5 of PREB. Adjust the mirror bias VR of the jig so that the error level is maximized and then measure the peak-to-peak value E_0 and the voltage Vbm being applied.
- Next, rotate the mirror bias VR all the way to the +12V side and measure the TRKG error p-p value, Ep. Then rotate the mirror bias VR all the way to the −12 V side and measure the TRKG error p-p value, En.

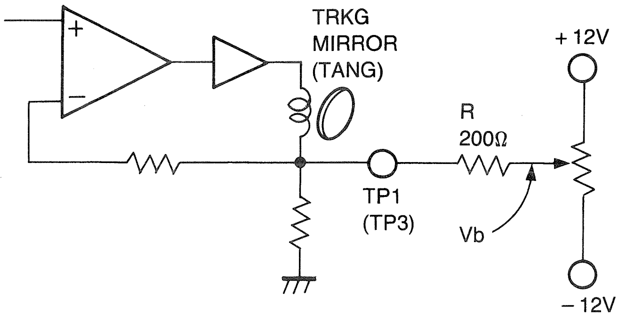


Fig. 11 Tracking Adjustment

VERIFYING OPTICAL AXIS IN TANG DIRECTION

- Connect the bias-voltage output terminal of the optical-axis checking jig to TP3 (TANG RTN) of SRVB.
- Measure the TRKG error level at TP5 of PREB. Adjust the mirror bias VR of the jig so that the error level is maximized and then measure the peak-to-peak value E_0 and the voltage Vbm being applied.
- Rotate the mirror bias VR all the way to the +12 side and measure the TRKG error p-p value, Ep.
  Then rotate the mirror bias VR all the way to the −12V side and measure the TRKG error p-p value, En.
- If Vbm is within the range of ±2.4V:
  $Ep > 0.63E_0$ and $En > 0.63E_0$
- If Vbm is outside the range of ±2.4V:
  $Ep > 0.70E_0$ and $En > 0.70E_0$
- If the above conditions are not satisfied, replace the pickup.

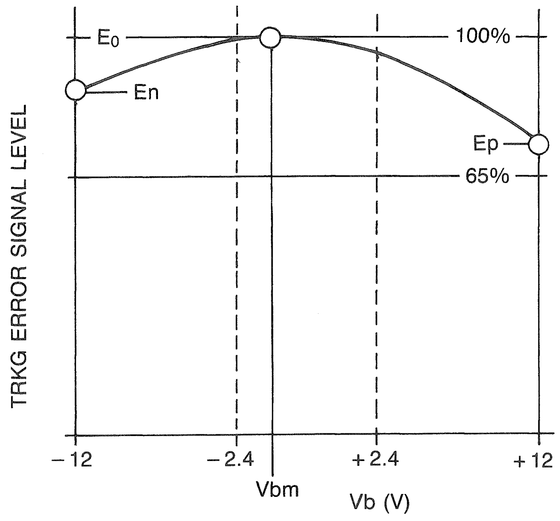


Fig. 12 Tracking Error Signal Level to Bias Voltage Vb

Good Pickups satisfy the following conditions:

(1) TRKG Error voltage becomes maximum when bias voltage decreases to 0V. — The beam is shooting the center of the mirror.
(2) The amplitude of the TRKG error signal is stable when a bias is applied to the mirror to turn it. — The mirror will deflect larger than the above and the tracking error level will decrease if the mirror's actuator is worn out. In other words, if the beam is not shooting the mirror center, the deflection of the mirror will be abnormal and the error voltage will decrease.

The following jigs are for checking the path easily.

* Optical Axis Check Jig (GGF-058) for LD-700 and CLD-900)
* VSOP Adj Box (GGV-112) and Harness GGV-111 for LD-1100 and -660

The VSOP Adj. Box can also be used to check the path of LD-700 and CLD-900 when a special connector is adopted. Their operating instructions have simplified descriptions of the above checking method. But VSOP adj. Box is no longer available.

3-3 FOCUS OFFSET

The above two methods are to verify that the reflected beam shoots the center of the Photo Diode. It is adjusted on X and Y axes in Fig. 13.

In adjusting Pickup, Z axis in addition to the X and Y axes should be imagined. The beam should be focused on the signal surface to read the pits. If the Objective-lens is off the right position, the beam will be beyond the control of FOCUS SERVO.

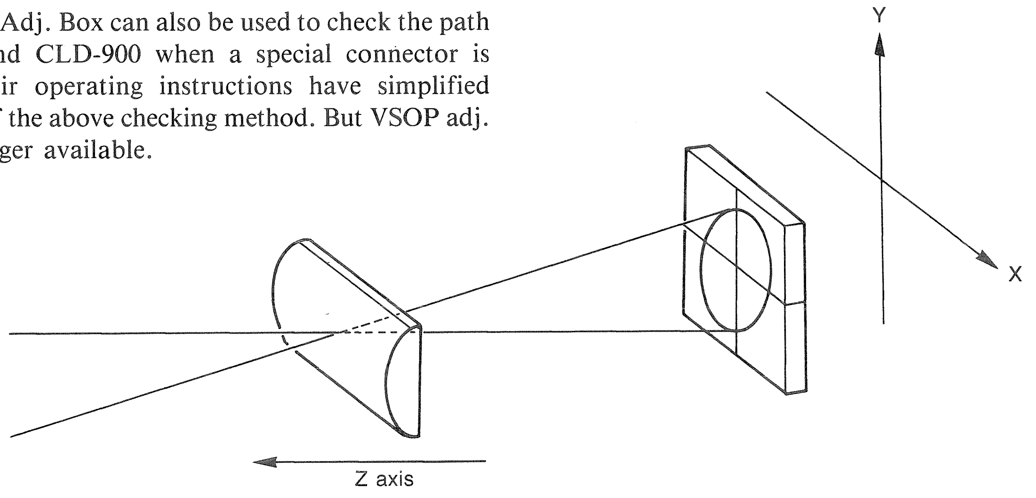


Fig. 13 Beam Spot on Photo-diode

a. Cylindrical-lens and Focus Servo

The laser beam reflected from the disc is round in section. The beam is deformed by the Cylindrical-Lens to make an oblique spot on the Photo-Diode (PD) when the beam is out of focus. This time the average distance between the Objective-lens and Cylindrical-lens becomes shorter or longer than the optimum, and the Focus Servo system can not keep the beam focused on the pitted signal surface.

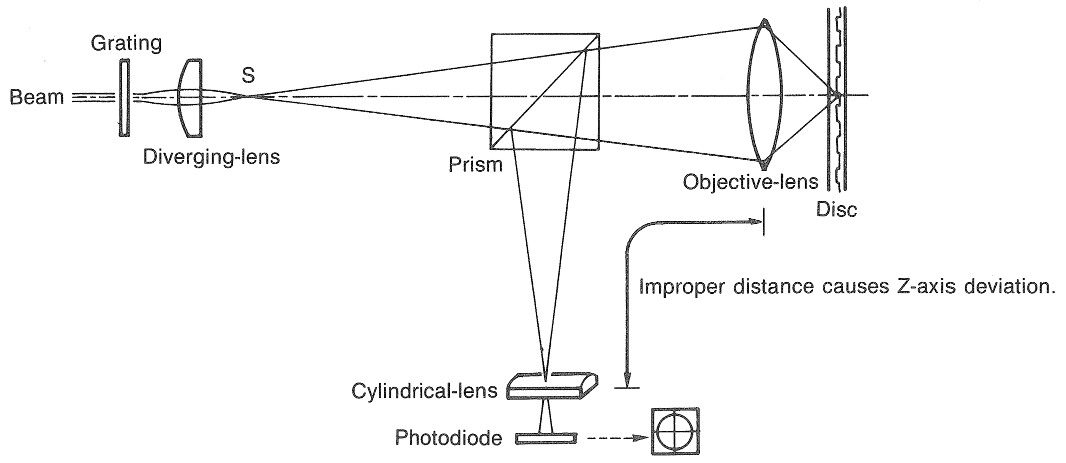


Fig. 14 Beam Path

When the distance between the Objective-lens and Cylindrical-lens is optimum: The FOCUS SERVO circuit moves the Objective-lens and decreases the Focus Error signal level down to 0V to keep the beam focused on the signal surface of the disc.

When it is improper: The reflected beam becomes round on the PD when the beam is out of focus on the disc. Then the FOCUS SERVO system tries to keep the Objective-lens out of focus and the beam spot on the signal surface is kept large causing crosstalk.

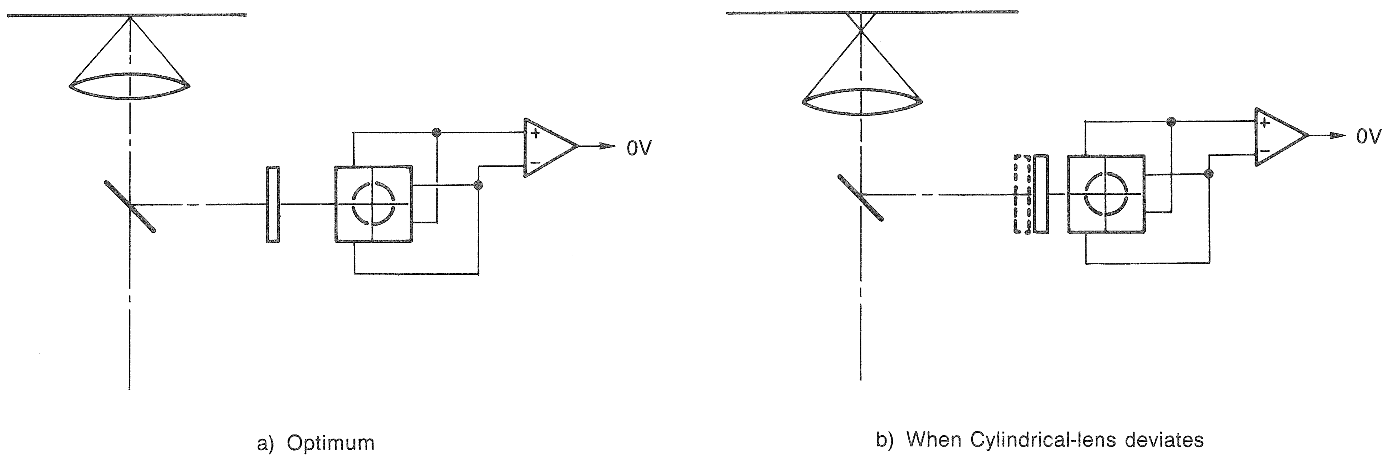


Fig. 15 Distance between Two Lenses

The Objective-lens, which is closer (farther) to Cylindrical-lens than the optimum, makes the spot on the PD oval when focused and causes it to generate a positive (negative) voltage. The beam can be kept focused on the signal surface leaving the spot on the PD oval by applying OP amplifier with a bias current for compensating or cancelling the error signal. However, this narrows the focus servo range and makes the beam hard to be refocused if it defocuses largely by scratches or dirt on the disc. Applying offset bias current is, therefore, not recommended. (CLD-900, LD-707, etc. have a Focus Balancer to solve the problem.) However, lens deviation on the Z-axis can be detected by applying this focus offset.

However, lens deviation on the Z-axis can be detected by applying this focus offset.

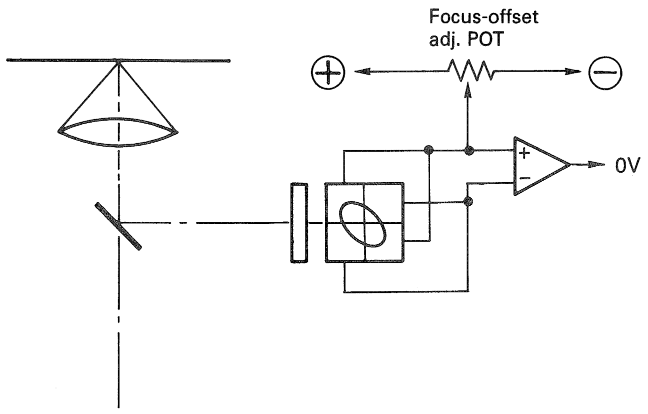


Fig. 15 c. Applying Bias Voltage

b. Checking the lens deviation on Z-axis

(1) Play a Test Disc.
(2) Adjust the FOCUS OFFSET POT to make the RF signal the maximum. Some units will become out of focus when you turn it excessively.
(3) Stop playing and measure the focus-error DC voltage in the STANDBY mode. The position of Pickup on Z-axis is optimum if the voltage is 0V. Replace Pickup if the voltage is out of the ±0.2V range.
(4) Finally, adjust the Focus-offset-adjustment-POT to make the focus error voltage 0V in STANDBY mode.

4. Mechanism Check

Now we can diagnose the Pickup. Mechanism and electronic circuits, however, should also be taken into consideration in diagnosing players because the Pickup is not working alone. Let's discuss the way of checking the mechanism. Mechanical deviation can be found visually. Electronic checkup is required to make it more accurate although our eyes are considerably accurate.

4-1 Non-aligned Motor

If the motor shaft is not aligned on the line on which Objective-lens travels, the symptom is the same as that of deviated Grating adjustment because the Tracking Servo is affected by it. In this case, after grating adjustment, the trouble is solved only at the Frame where it is adjusted, and the grating deviation increases as the Pickup moves far from the Frame. Grating adjustment is usually made in the midway on the disc. TRACK SKIPPING or NO PLAY will, therefore, result on inner or outer tracks.

Motor shaft alignment can be checked by comparing Lissajous figures of TRKG (A-B) and TRKG (A + B) keeping the tracking servo loop open as in Grating adjustment. The figures should stay unchanged on inner and outer tracks.

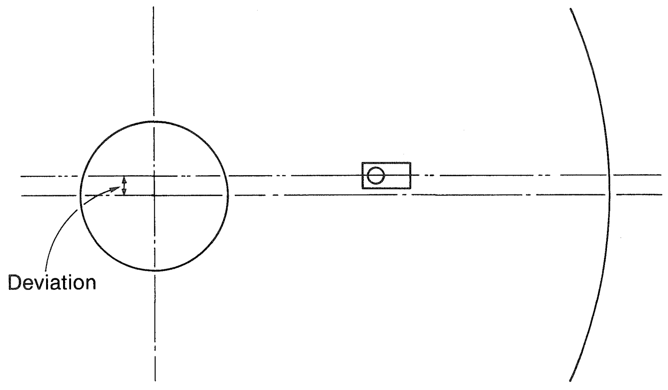


Fig. 16 Non-aligned Motor

A simple checking method

(a) Connect TRKG (A-B) and TRKG (A + B) to X and Y channels of an oscilloscope respectively.
(b) SEARCH throughout a disc from the inmost track to the outmost or vice versa.
(c) Observe the Lissajous figures while searching because Tracking Servo opens and the figures may vary this time. If it is not a horizontal line, grating should be adjusted. If it grows thick on inner or outer tracks, the motor should be realigned.
Note: A CAV disc available on the market is recommended for checking motor alignment because the track pitch of an F-series test disc differs by frame-ranges and the thickness of the Lissajous figures normally shifts as the Pickup travels as shown in Fig. 17. With the CAV disc available on the market, the alignment is all right as long as the figure does not change although it may be thick. If the motor is deviated, adjust Grating on tracks midway or farther and then adjust the motor position on inner tracks observing Lissajous figures.

In case you use an F-series test disc, to avoid misadjustment, adjust the Grating on outer tracks by making the Lissajous figures a vertical line (on-track state*) and then adjust the motor position on inner tracks so that it becomes a vertical line, and finally readjust the Grating to make it a horizontal line to get an optimum Grating angle. Refer to "LD-motor position" of the service manual of CLD-909 for the details.

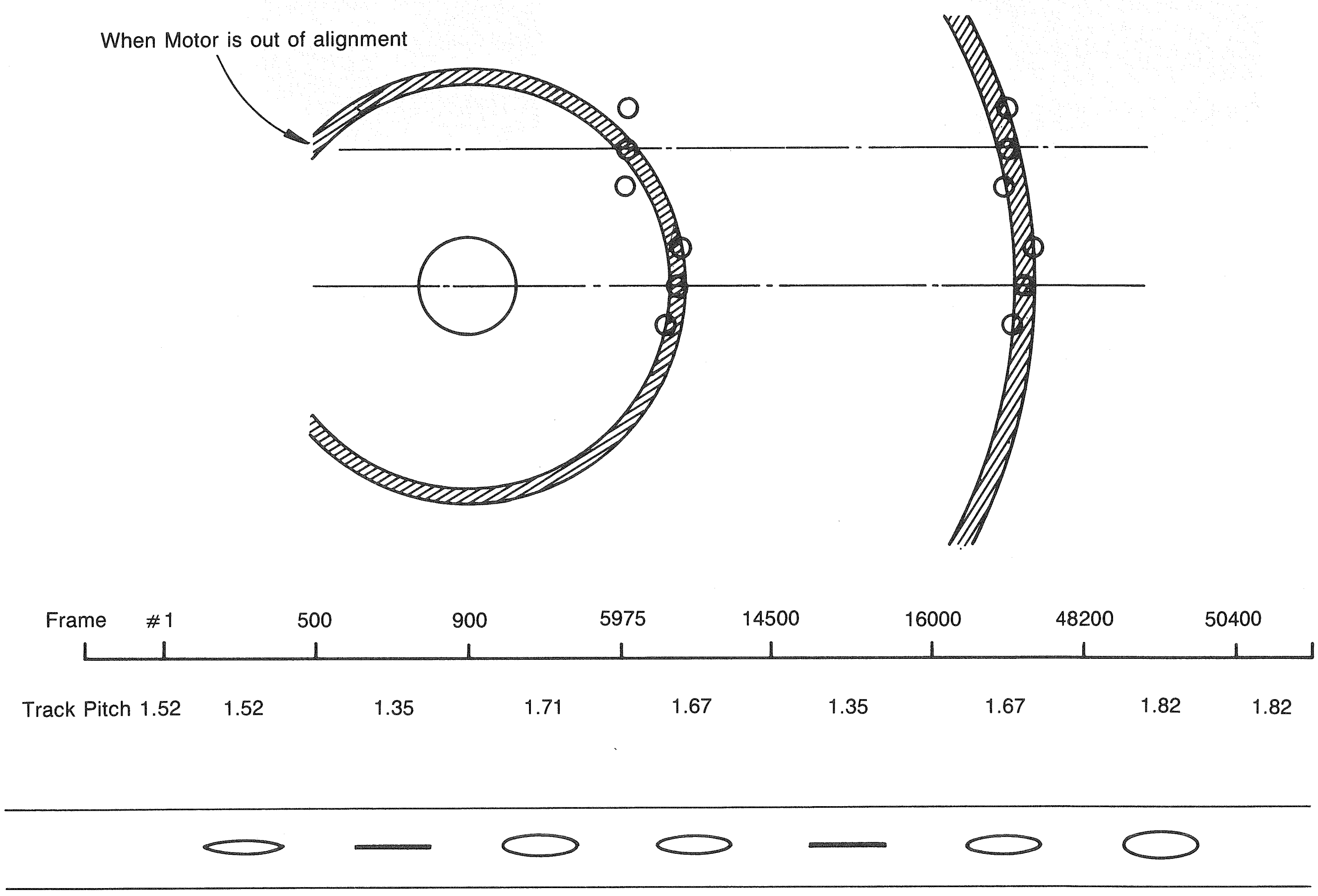

When Motor is out of alignment
Frame #1 500 900 5975 14500 16000 48200 50400
Track Pitch 1.52 1.52 1.35 1.71 1.67 1.35 1.67 1.82 1.82

Fig. 17 Lissajous Figures Shifts As Pickup Travels

Note: ON TRACK

When adjusting the Grating, open Tracking Loop and find out the position where the envelope of TRKG error signal becomes clear and where the amplitude becomes the minimum by turning Grating adjustment screw. This time the three beams trace the center of the same track and the Lissajous figures become a vertical line. The state and position are called "ON TRACK" and "NUL POINT" respectively.

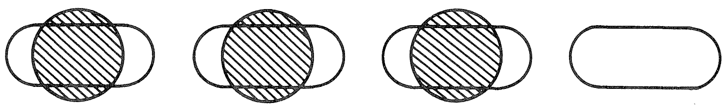
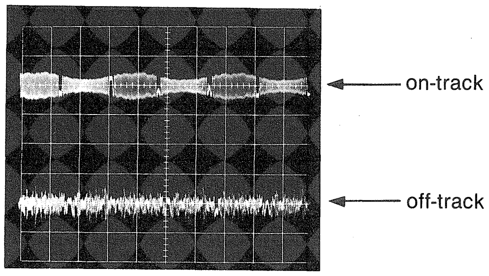
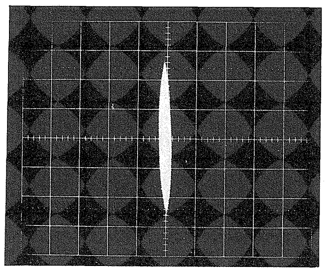


Fig. 18 ON-TRACK state



5 mS/Div 100 mV/Div

Photo 6 TRKG Error



Xch: TRKG error signal
Ych: TRKG (A + B) signal

A: T1 — F — T1' X: 20 mV/Div
Y: 20 mV/Div

Photo 7 Lissajous Figures in On-track state

4-2 Tilted Spindle-motor from the tracking direction

What will happen if the motor is out of perpendicular to the locus of Pickup? The symptoms are the same as those when playing a warped disc. In case of LD-707 and CLD-909, this trouble occurs if the Pickup is not parallel to the Slider-base. The symptoms are:

* Crosstalk appears.
* Focus Lock becomes difficult.
* Pickup goes out of focus as it moves outward.

The players which have a Tilt Servo system can keep Pickup parallel to the disc to some extent. The distance between them, however, can not be kept constant.

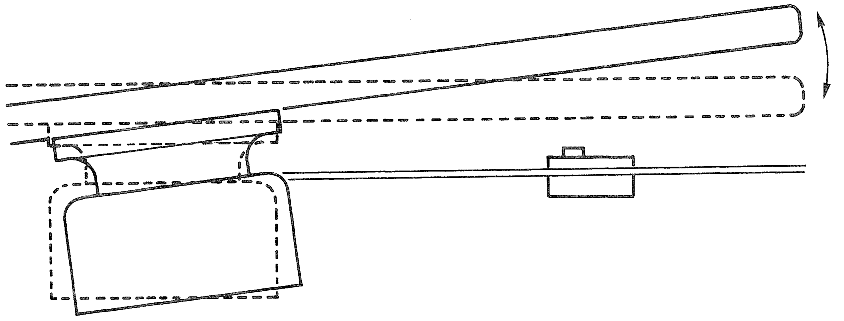


Fig. 19 Tilted Motor from TRKG Direction

Checking the tilt of Pickup or Spindle-motor

* Observe Focus-return voltage filtered by LPF on an oscilloscope.
* Verify that the DC voltage does not vary on inner and outer tracks.
* In case of 12″ disc, check it on midway and inner tracks of a less warped disc. If there is a difference of more than 0.1V in FOCUS-Return voltage between the readings on inner and center (midway) tracks, the Motor or Pickup can be considered to be tilted.

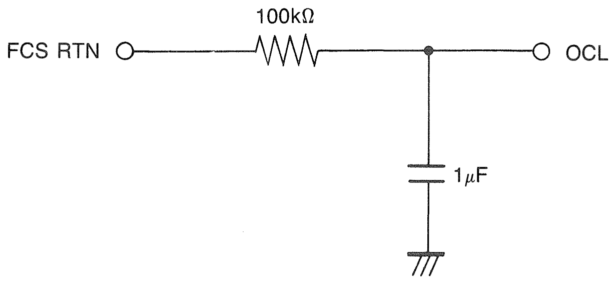


Fig. 20 LPF

Test Point

Models which have a series circuit of Focus-coil and Return-resistor such as LD-1100, LD-700, P-D70, etc.

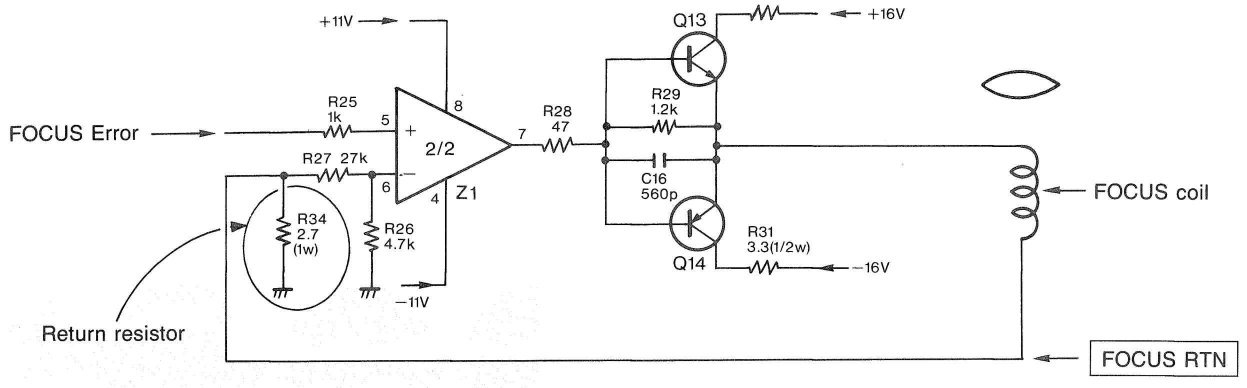


Fig. 21 Test Point of LD-1100, LD-700, P-D70, etc.

Models which have no Return-resistor (grounded Focus-coil) such as LD-707, CLD-909, etc.

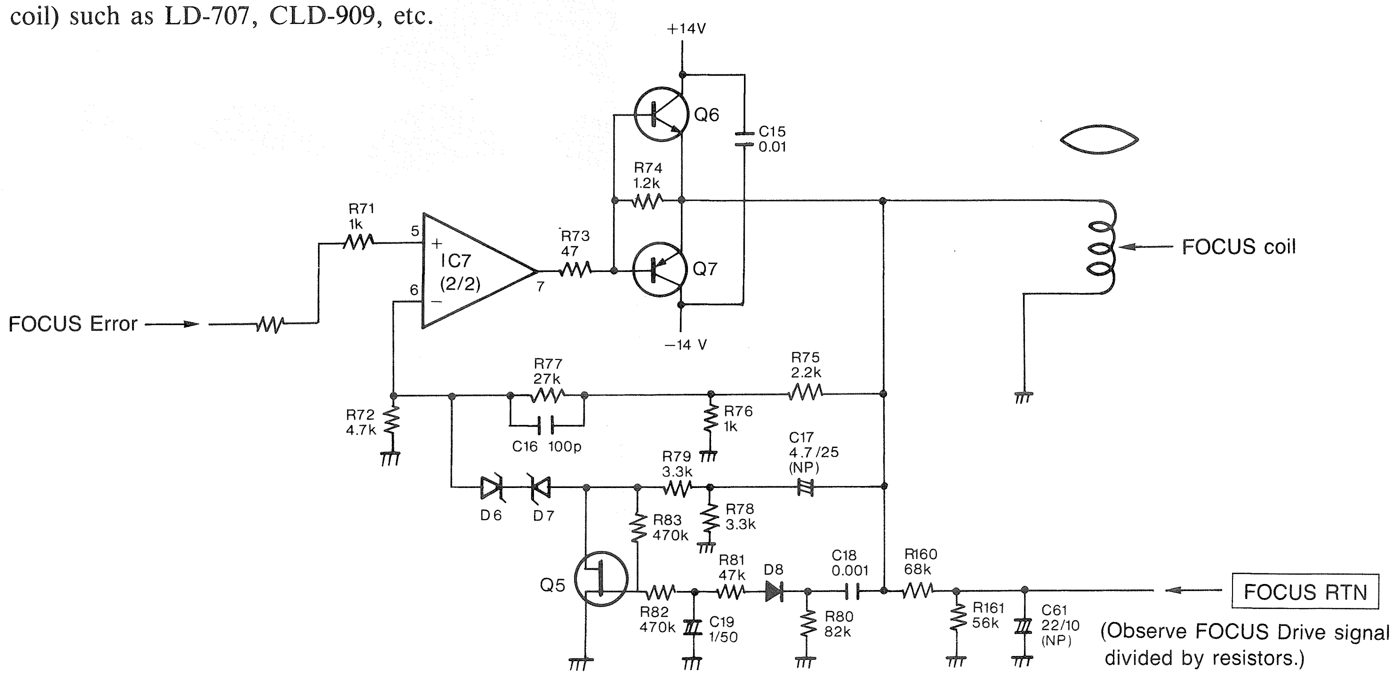


Fig. 22 Test Point of LD-707, CLD-909, etc.

4-3 Tilted Spindle-motor from the tangential direction

What will happen when the Motor or Pickup tilts from the tangential direction?
LD player: Crosstalk — Adjust the perpendicularity of Motor or Pickup to obtain the maximum RF signal.
CD player: Dropout — Observe RF (EFM) signal waveform at the infinity-0 point on TRK #23 of Test Disc (YEDS-7). Adjust them to make the top and bottom portions of the trace line horizontal on the oscilloscope.

On the models which generate obscure waveforms such as PD-5010 series, judge it observing the jitter component of EFM signal.

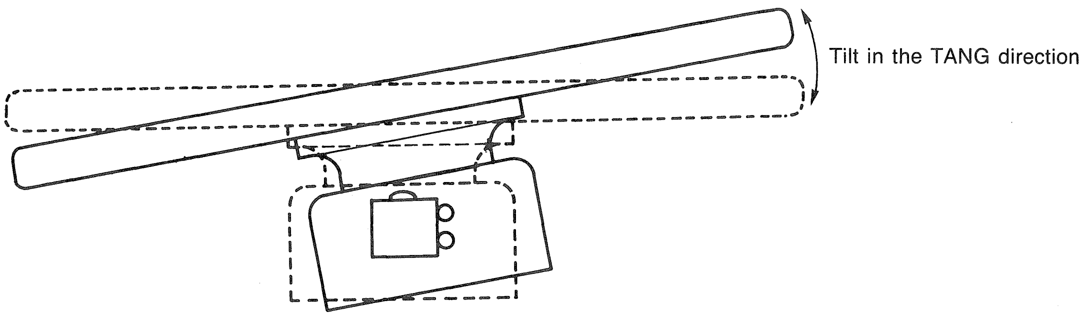
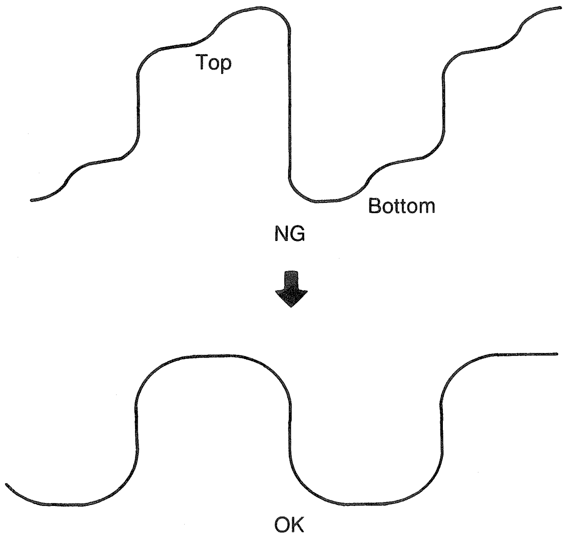


Fig. 23 Tilted Motor From TANG Direction



Tilt in the TANG direction

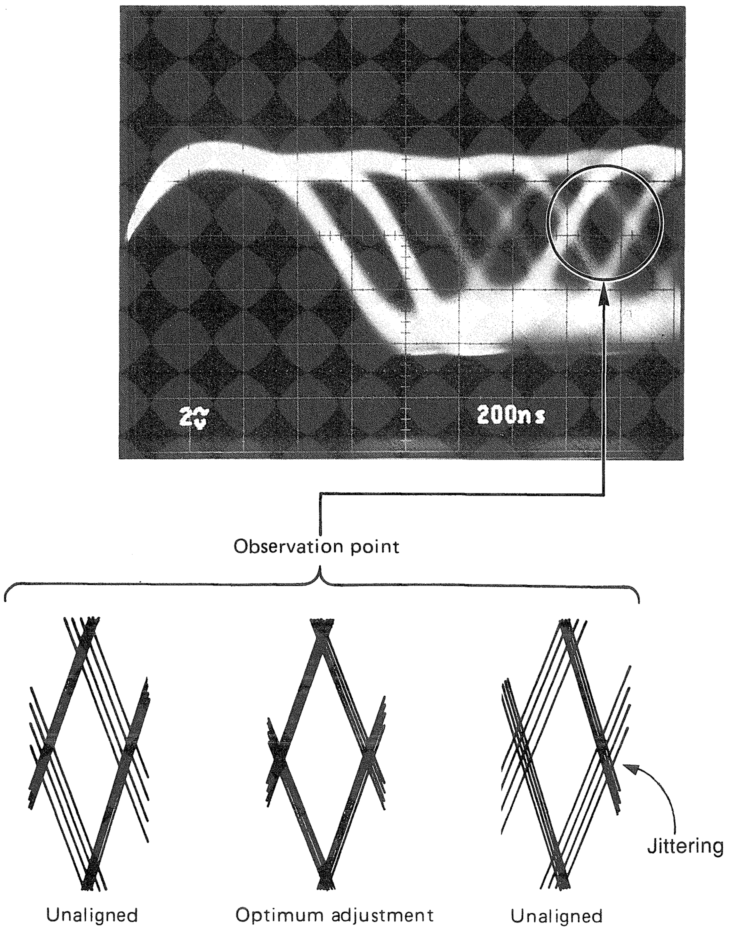


Photo 8 EFM Signal

5. Eccentricity

Every disc and motor has eccentricity to some extent. It is compensated by TRKG and TANG servo systems and CCD (Charge-Coupled Device). The beam skips tracks or SPDL Lock is released when the eccentricity is too large.

The degree of the symptom varies every time the disc is reloaded because it depends on the clamping conditions. Try to take the disc out, turn it by 90° and load it again. The causes of the eccentricity are:

* Worn Clamper or Clamper-bearing
* Faulty peripheral component of Clamper
* Barrs or wearing of the center hole of the disc

If the player sometimes skips, find out whether the player is eccentric or Pickup itself is faulty by the following method.

Measuring method of disc eccentricity

1. Play a disc, and open the TANG SERVO (or CCD SERVO) loop. To open the loop, refer to the following page and the adjustment process in Service Manual.
2. Observe the trapezoid waveform signal for generating TANG (CCD) Error, and measure the jittering period on an oscilloscope.

Jitter time	Eccentricity
Less than 20μsec	Normal
20 — 30μsec	A little large
30μsec or more	Causes skipping

Note: With a CAV disc measure the jitter time on inner tracks (between frames #1 and #5,000). The time becomes shorter as the Pickup moves outward. With a CLV disc, the jitter time is almost constant irrespective of tracking position.

3. If the measurement results are excessive with every scratchless disc, LV players may be faulty due to the faulty SPDL-motor shaft, clamper, or faulty clamper-shaft-holder.

The arrows in the Fig. 25 and 26 show the test points for this measurement. As for other models, refer to the respective service manual and measure it in the same way as this. In case of LD-700, ground the TP7 to open the Tangential Servo and observe the signal at Z202 (PA9002)-1 on the oscilloscope. In case of LD-707, open CCP Servo by grounding IC3 (PA0017)-9 and observe it at IC7 (PA5009)-9.

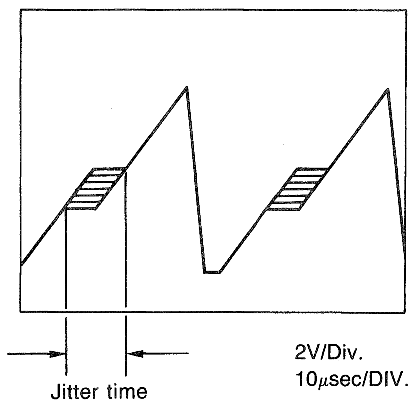


Fig. 24 Jitter Time

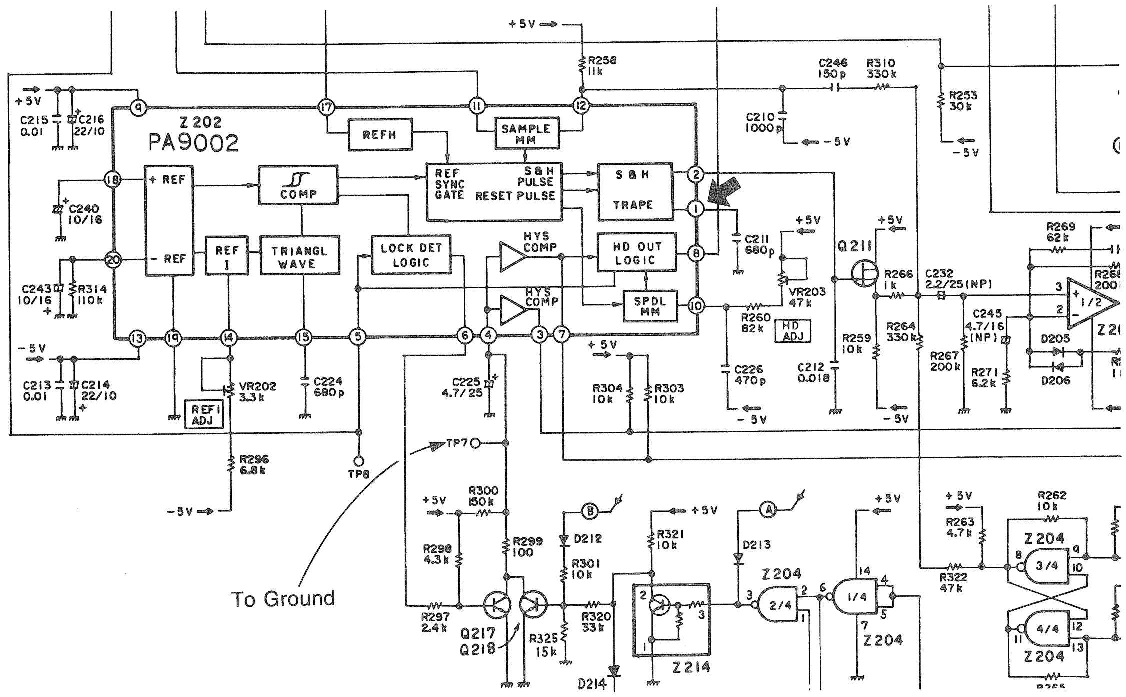


Fig. 25 Measuring Jitter-time of LD-700

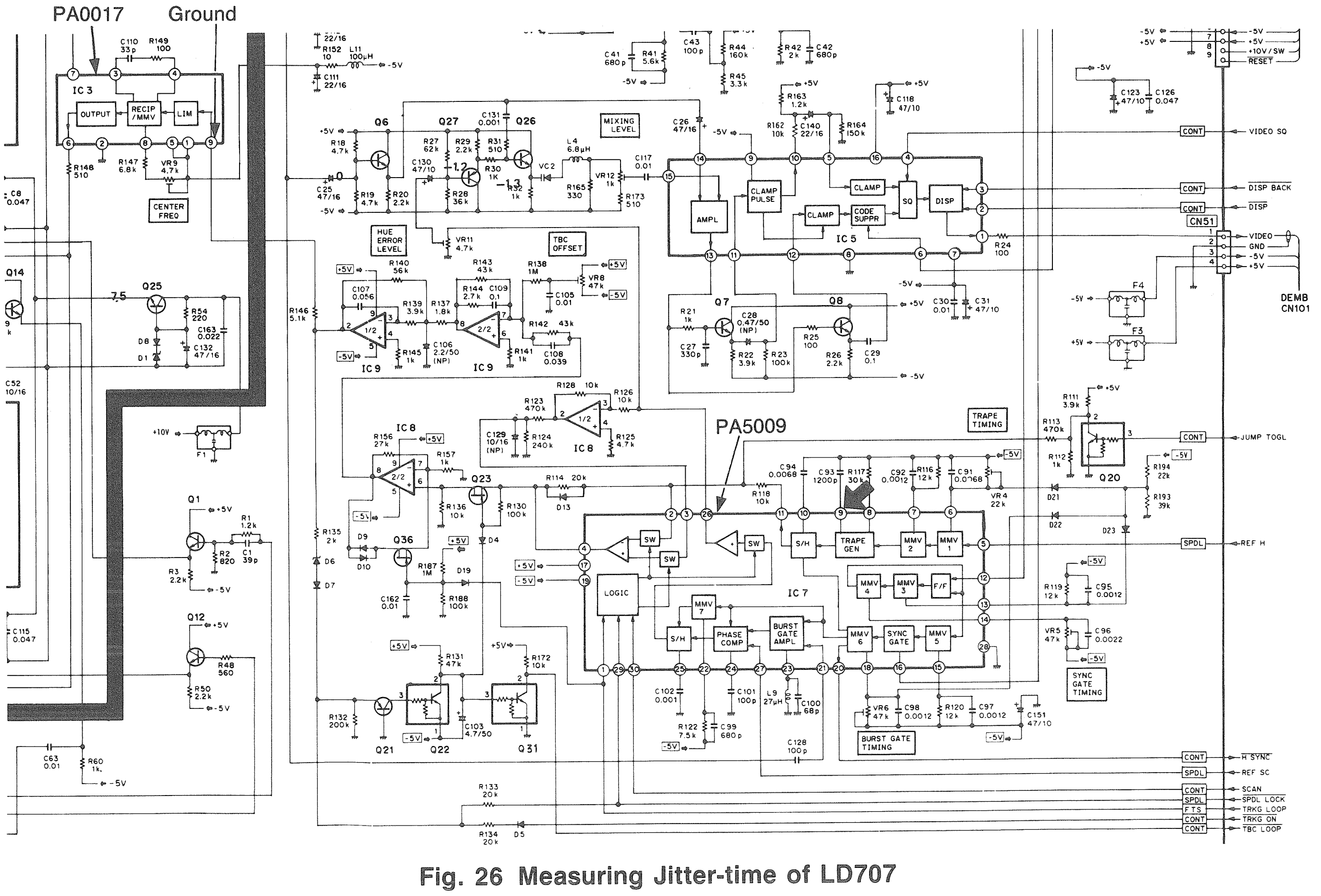


Fig. 26 Measuring Jitter-time of LD707

6. Overall checkup of Optical Pickup System

Let's summarize the check points.

a. Spindle-motor does not turn when a disc is loaded and PLAY is pressed.

Laser Diode does not glow so bright as He-Ne tube due to it's wave length. If a part of Objective-lens is bright in red, the Laser Diode is on.

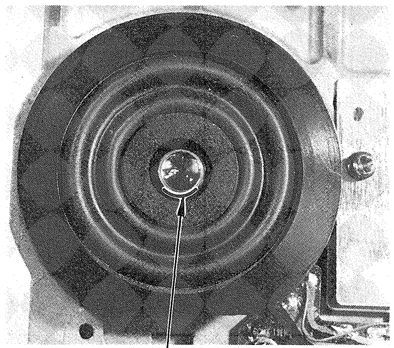
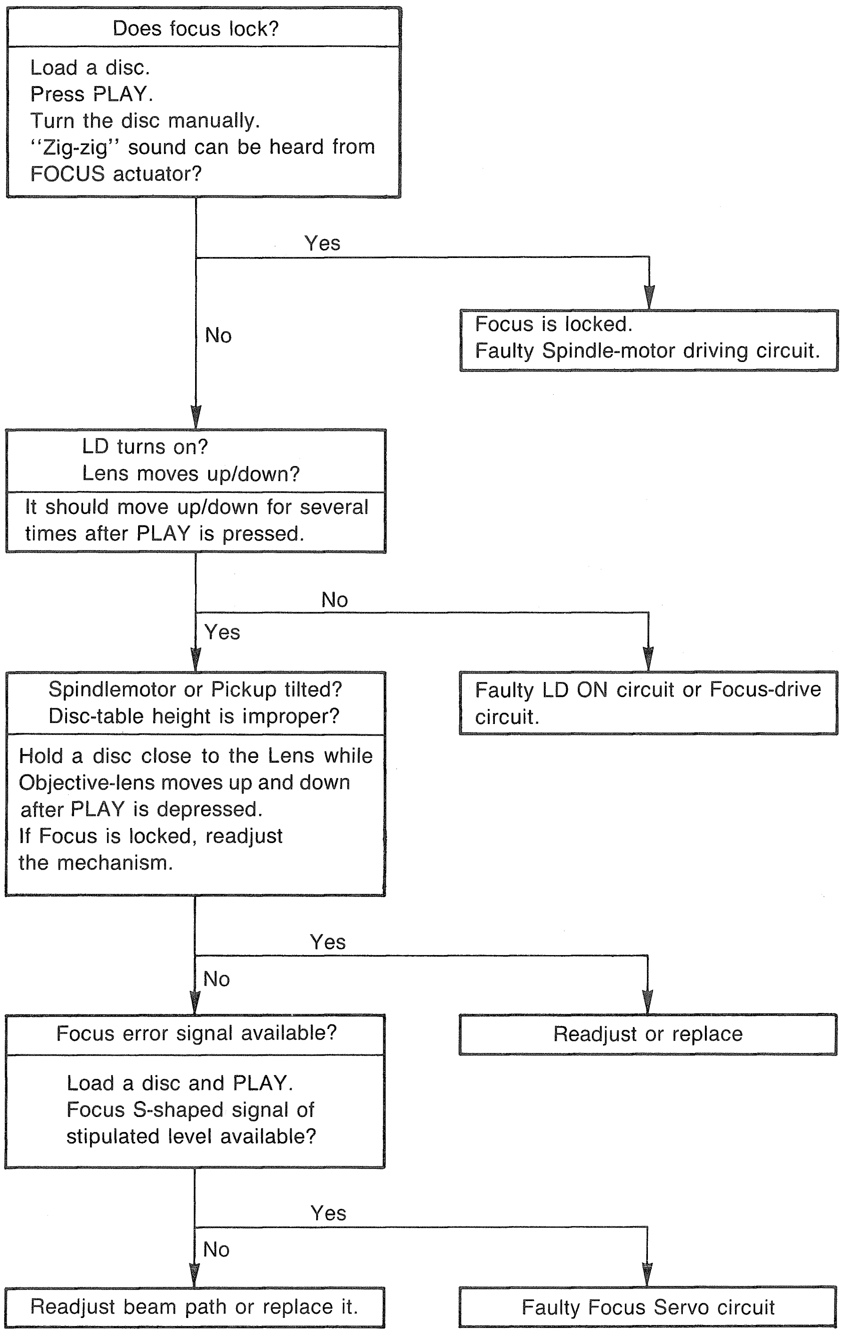


Photo 9 FOCUS (Objective) Lens



Does focus lock?

Load a disc.
Press PLAY.
Turn the disc manually.
"Zig-zig" sound can be heard from FOCUS actuator?

Yes → Focus is locked. Faulty Spindle-motor driving circuit.

No ↓

LD turns on?
Lens moves up/down?

It should move up/down for several times after PLAY is pressed.

No → Faulty LD ON circuit or Focus-drive circuit.

Yes ↓

Spindlemotor or Pickup tilted?
Disc-table height is improper?

Hold a disc close to the Lens while Objective-lens moves up and down after PLAY is depressed.
If Focus is locked, readjust the mechanism.

Yes → Readjust or replace

No ↓

Focus error signal available?

Load a disc and PLAY.
Focus S-shaped signal of stipulated level available?

Yes → Faulty Focus Servo circuit

No ↓ Readjust beam path or replace it.

Fig. 27 Checking Pickup and Motor Mechanism

b. Spindle-motor starts rotating after a Disc is loaded and PLAY is pressed. (SPDL SERVO does not lock but skips, etc.)
Note: If it stops soon but keeps rotating when Tracking Servo is opened, check it on the four following items keeping the servo opened.

Table Checking Beam Path

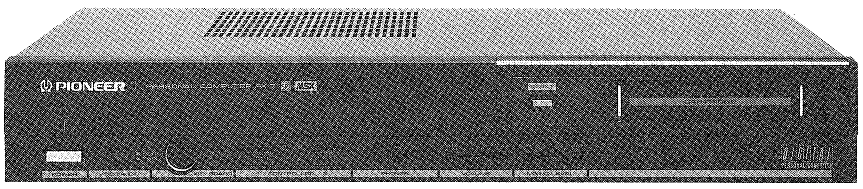
Tracking Servo Loop	Observation Points	Items
Open	X-ch: TRKG (A – B) Y-ch: TRKG (A + B)	Grating Play a Test-disc. Lissajous figures become a horizontal line at the prescribed Frame
Open	TRKG (A – B)	TRKG Balance TRKG (A – B) signal center: 0V?
Open	X-ch: TRKG (A – B) Y-ch: TRKG (A + B)	SPDL motor alignment Lissajous figures should not grow fat when searched in both directions.
Open or Close	FOCUS RTN through LPF	Tilt in TRKG direction DC voltage should be stable when Pickup is moved to inner or outer tracks.
Close	RF	Tilt in TANG direction CD: Top and bottom portions of RF (EFM) signal should be flat. LD: Amplitude of RF signal should be max.

By K. Nohara

Microcomputer Fundamentals

The microcomputer is a tiny computing system. The equipment itself is called "hardware". Program instructions or the way of using it is called "software". The software permanently stored in a ROM (read only memory) is called firmware. Its main blocks have been integrated into one or a few LSI's. The difference from the minicomputer and the larger computer is in size, speed of execution, computing power and price. The miniaturizing technique of electronic circuits has given the microcomputer an ability close to that of old large computers.

The microprocessor, or central processing unit (CPU) which is the brain of the microcomputer, is being employed in various equipment and controls their functions. The knowledge of computers is usually unnecessary for our repair service because most operations are performed inside the LSIs and the CPU can be considered to be a black box. Troubleshooting those equipment, however, will be easy if you have the knowledge. Here we will discuss the fundamental principle of the microcomputer basing on the Zilog Z80ACPU. Refer to its instruction manual for details. Photo 1 shows our PX-7, MSX Personal Computer in European version which employs a Z80A.



PX-7, MSX Personal Computer in European Version

Tablet PX-TB7

Photo 1

1. Number Systems

Digital computers use the binary system although we are accustomed to the decimal system. A digital signal has only two states, low (L) and high (H) or 0 and 1. They also use octal and hexadecimal systems because the unit of the processor in computers is 4 and 8 bits. Table 1 shows their differences. As for the details, refer to the "DIGITAL — Supplement of TUNING FORK".

Table 1 Five Numbers Systems

Decimal	Binary	Octal	Hex	BCD
0	0	0	0	0000 0000
1	1	1	1	0000 0001
2	10	2	2	0000 0010
3	11	3	3	0000 0011
4	100	4	4	0000 0100
5	101	5	5	0000 0101
6	110	6	6	0000 0110
7	111	7	7	0000 0111
8	1000	10	8	0000 1000
9	1001	11	9	0000 1001
10	1010	12	A	0001 0000
11	1011	13	B	0001 0001
12	1100	14	C	0001 0010
13	1101	15	D	0001 0011
14	1110	16	E	0001 0100
15	1111	17	F	0001 0101
16	10000	20	10	0001 0110
17	10001	21	11	0001 0111
18	10010	22	12	0001 1000
19	10011	23	13	0001 1001
20	10100	24	14	0010 0000

There are several ways to represent data using the binary code such as straight binary, signed binary, 2's complement, binary coded decimal (BCD) and American Standard Code for Information Interchange (ASCII).

1.1 Straight Binary

In unsigned representation, an n-bit binary data word is simply used to represent the first 2^n nonnegative integers. An 8-bit data word can represent the integers 0 through 255 as shown on Table 2.

Table 2 Straight Binary Representation

Straight binary representation	Decimal numbers
00000000	0
00000001	1
00000010	2
00000011	3
00000100	4
⋮	⋮
11111111	255

1.2 Signed Binary

Representation of both positive and negative numbers with the straight binary is impossible. The signed binary spares the most significant bit (MSB) for representing the sign of the numbers. The lower-order bits are used to represent the absolute value in straight binary as shown in Table 3. This is seldom used for two reasons. First, there are two representations for the number zero. Second, the ordinary rules of addition for straight binary cannot be applied to signed binary numbers.

Table 3 Signed Binary Representation

Signed binary representation	Decimal numbers
011111111	+127
011111110	+126
⋮	⋮
000000010	2
000000001	1
000000000	0
100000000	0
100000001	−1
100000010	−2
⋮	⋮
111111110	−126
111111111	−127

1.3 BCD

Each decimal digit can be represented by a combination of four binary numerals. 4 bits can represent 16 codes. 10 codes from 0000 to 1001 among the 16 codes are used for binary representation. Other combinations are not used in this system. This is useful for interfacing numeric keyboards and displays. Refer to Fig. 1.

Examples

3	9	Decimal (D)
0011	1001	Binary (B)
6	5	Decimal
0110	0101	Binary

Fig. 1 BCD Representation

1.4 2's complement

2's complement notation is most commonly used in microcomputers for performing arithmetic with signed numbers. In this notation, the negative of a binary number is represented as its 2's complement. A complement of a number is formed simply by changing the 1's to 0's and the 0's to 1's. The 2's complement of a number is formed by adding one to the complement of the number.

Example		
Straight binary	0 0 0 0 1 0 1 0	10 in decimal (D)
Changing	1 1 1 1 0 1 0 1	Complement
+ 1	1 1 1 1 0 1 1 0	2's Complement

As mentioned 256 different numbers can be represented by 8 bits. The 0 on the MSB represents a positive number and 1 represents a negative one. If 011111111 represents +127 and is counted down, 0, −1 and −128 become 00000000, 11111111 and 10000000 respectively.

The numbers from +127 to −128 are available by decrimenting or incrementing throughout the range.

Table 4 2's Complement

Twos complement	Decimal number
0 1 1 1 1 1 1 1	127
0 1 1 1 1 1 1 0	126
⋮	⋮
0 0 0 0 0 0 1 0	2
0 0 0 0 0 0 0 1	1
0 0 0 0 0 0 0 0	0
1 1 1 1 1 1 1 1	−1
1 1 1 1 1 1 1 0	−2
1 1 1 1 1 1 0 1	−3
⋮	⋮
1 0 0 0 0 0 0 1	−127
1 0 0 0 0 0 0 0	−128

1.5 ASCII

In ASCII notation 7 bits of one byte code are used for representing decimal numbers, alphabetical characters and machine control commands. The MSB of the code is used for parity or error testing. The seven bits provides 128 possible characters. Decimal numbers are represented as Table 5a. 32 combinations are assigned to control commands which are not printed out but provided to handle hardware operations.

Table 5a Decimal Number to ASCII Codes

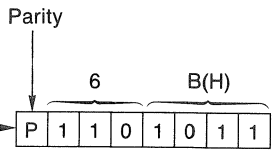
P	ASCII code	Decimal number
	0 1 1 0 0 0 0	0
	0 1 1 0 0 0 1	1
	0 1 1 0 0 1 0	2
	0 1 1 0 0 1 1	3
Parity	0 1 1 0 1 0 0	4
	0 1 1 0 1 0 1	5
	0 1 1 0 1 1 0	6
	0 1 1 0 1 1 1	7
	0 1 1 1 0 0 0	8
	0 1 1 1 0 0 1	9

$\underbrace{\quad}_{3}\ \underbrace{\quad}_{0\sim 9}$

Table 5b ASCII Codes Represented byHexadecimal Numbers

Hex	Symbol	Hex	Symbol	Hex	Symbol	Hex	Symbol
00	NUL	20	SPACE	40	@	60	.
01	SOH	21	!	41	A	61	a
02	STX	22	"	42	B	62	b
03	ETX	23	#	43	C	63	c
04	EOT	24	$	44	D	64	d
05	ENQ	25	%	45	E	65	e
06	ACK	26	&	46	F	66	f
07	BEL	27	'	47	G	67	g
08	BS	28	(	48	H	68	h
09	HT	29	)	49	I	69	i
0A	LF	2A	*	4A	J	6A	j
0B	VT	2B	+	4B	K	(6B	k)
0C	FF	2C	,	4C	L	6C	l
ID	CR	2D	—	4D	M	6D	m
0E	SO	2E	.	4E	N	6E	n
0F	SI	2F	/	4F	0	6F	o
10	DLE	30	0	50	P	70	p
11	DC1	31	1	51	Q	71	q
12	DC2	32	2	52	R	72	r
13	DC3	33	3	53	S	72	s
14	DC4	34	4	54	T	74	t
15	NAK	35	5	55	U	75	u
16	SYN	36	6	56	V	76	v
17	ETB	37	7	57	W	77	w
18	CAN	38	8	58	X	78	x
19	EM	39	9	59	Y	79	y
1A	SUB	3A	:	5A	Z	7A	z
1B	ESC	3B	;	5B	[	7B	|
1C	FS	3C	<	5C	\	7C	|
1D	GS	3D	=	5D	]	7D	|
1E	RS	3E	>	5E	∧	7E	~
1F	US	3F	?	5F	-	7F	DELETE

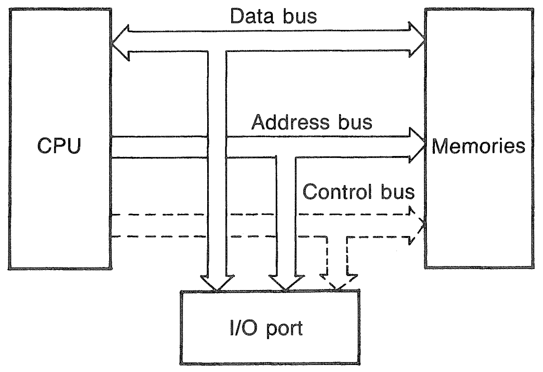
Codes in squares are for operating hardware



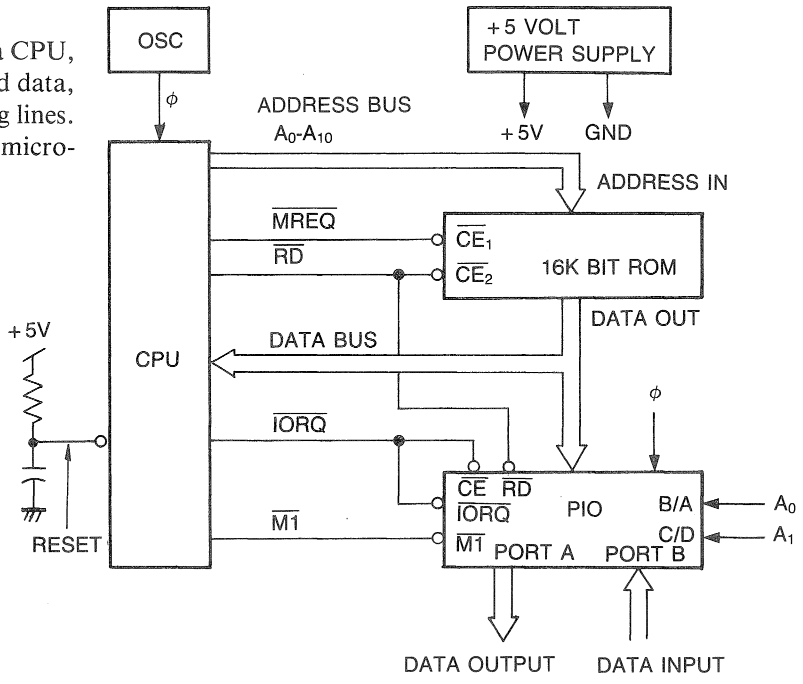
LF =	Line Feed
FF =	Form Feed
CR =	Carriage Return
CTRL =	Control Character
BS =	Backspace
HT =	Horizontal tab
VT =	Vertical tab
DEL =	Rub out

2. Architecture

The hardware of the microcomputer consists of a CPU, memories or information storages for programs and data, input/output interfaces and buses or communicating lines. Fig. 2 shows simplified block diagrams of the microcomputer.



(a) The Simplest Diagram



(b) Small Computer System Using a Z80ACPU

Fig. 2. Computer System

2.1 Memories

Microprocessor systems use binary memories to store programs and data. Now a 1/3-square-inch chip of a memory device has a capacity of over 65k bits. It can store more than 8 thousand alphanumeric characters. Each of the memories has its own address. The address decoder on the memory chip decodes the address data and connects the selected memory to the data pins. Fig. 3 shows a simplified diagram of an 8-bit memory.

ROM: The information is stored permanently or semipermanently and is read out, but not altered in operation.

RAM (Ramdom-access Memory): A Read/Write memory into which data can be stored and then retrieved.

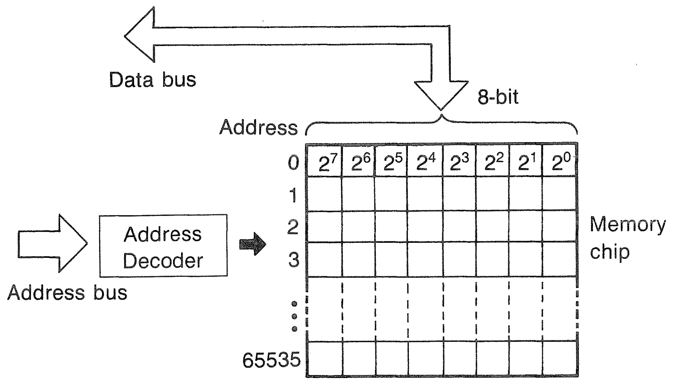
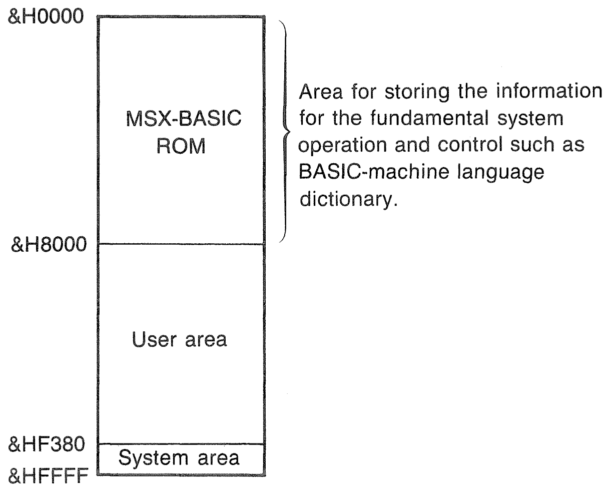


Fig. 3 8-bit Memory Device



&H0000
MSX-BASIC ROM
Area for storing the information for the fundamental system operation and control such as BASIC-machine language dictionary.
&H8000
User area
&HF380
System area
&HFFFF

Fig. 4 Simplified Memory Map of PX-7

The principle of ROM

One of the simple memory devices is a transister. A transistor can memorize one bit of information by keeping its base open or closed. Fig. 5 shows the magnified diagram of the FUSE ROM. The eight transistors hanging on the same drive line consist a set of memories that has one address. A set of binary information (11001010) is available at the output when the power V is supplied to the circuit and the addressed drive line (B) is made H, because the source current flows only through the transistors connected at the emitters, and the level on the output lines disconnected from the transistors stays L. Other set of transistors are kept inactive because their drive lines are kept L.

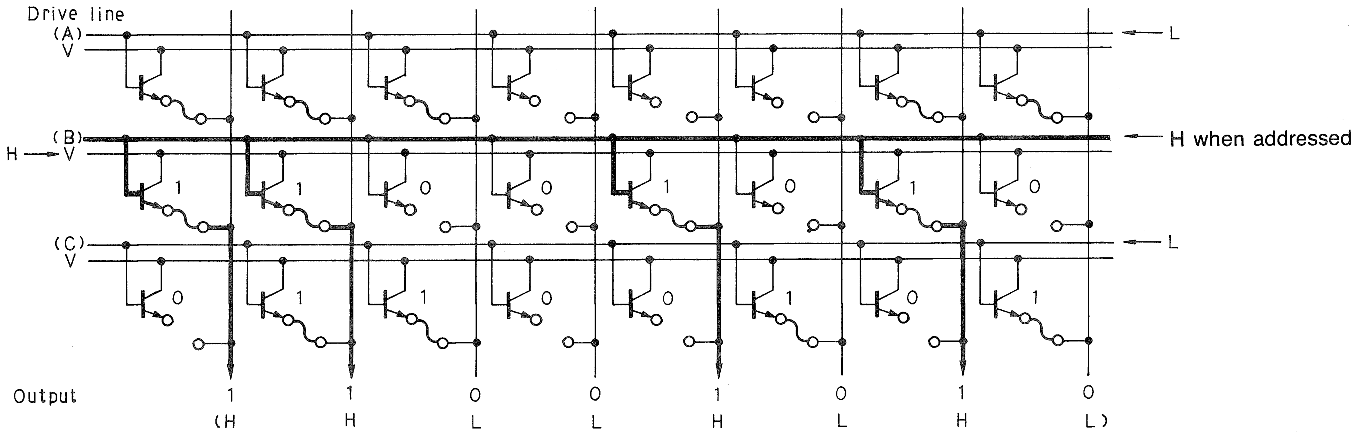


Fig. 5 Principle of ROM

Fig. 6 shows a ROM that has sixteen 1-bit memories. The data of memory #5 appear at the output when X_1 and Y_1 are closed. Usually there are eight bits in one memory unit and eight lines for the output.

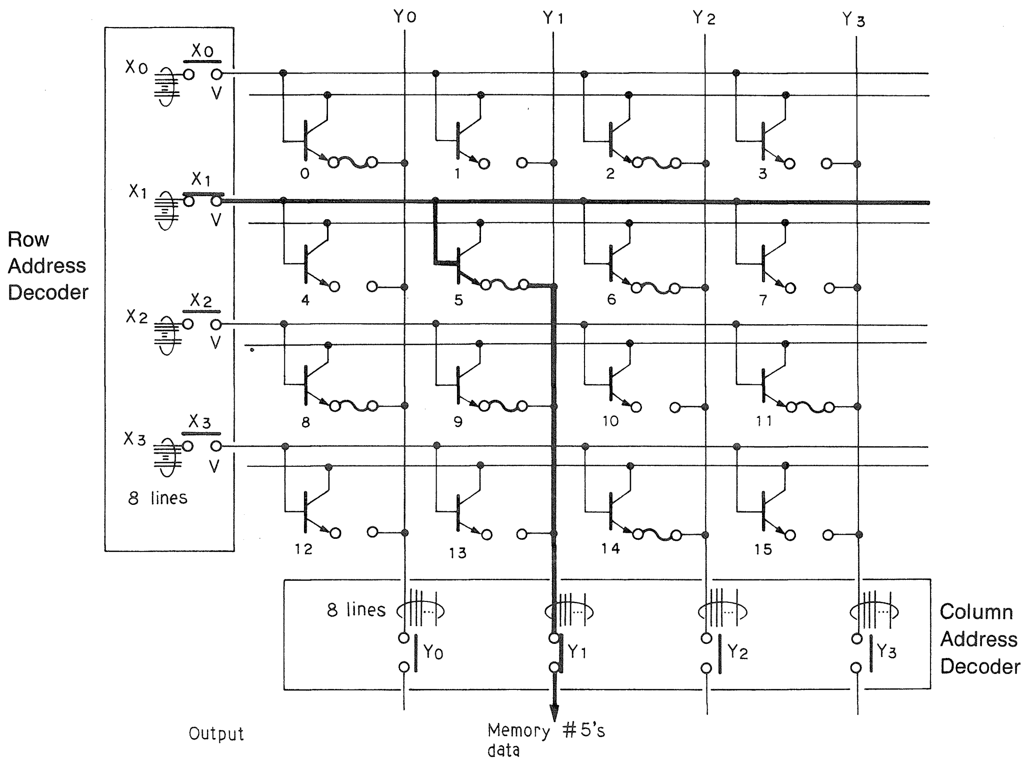


Fig. 6 Addressing Memory Matrix

2.2 CPU

Fig. 7 shows the simplified block diagram of the CPU. It receives data from memory and external devices, processes them and puts out the result controlling the operation of the whole system.

The CPU can perform a great variety of tasks depending on the software. However, it just reads instructions that come from memories or external devices and executes them. Those appliances of particular purposes such as washing machines are not computers even they employ microprocessors because they are operated only by a few nonflexible programs.

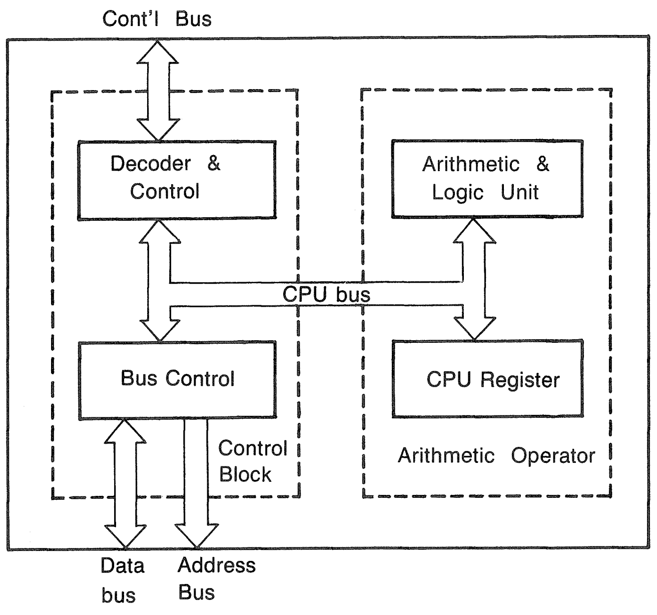
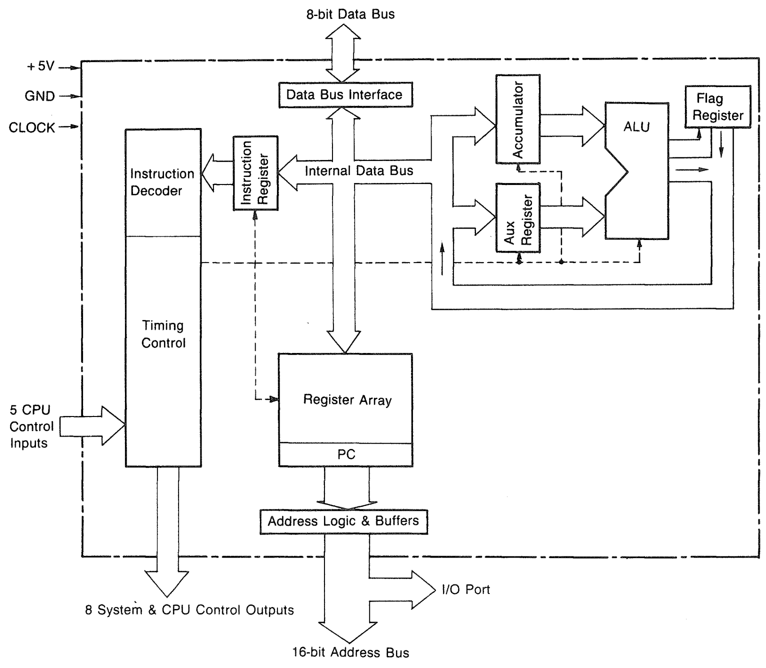


Fig. 7 Simplified Block Diagram of CPU

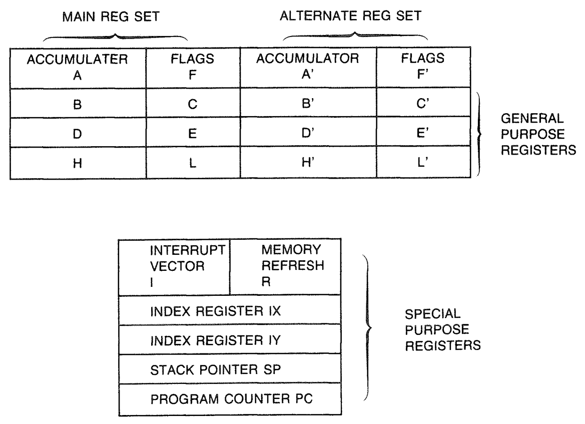
Devices in CPU

ALU (Arithmetic and Logical Unit)

This is an adder made of logic circuits such as AND, OR and Exclusive-OR and performs arithmetic and logical operations supplemented by shifting and sequencing circuitry for implementing multiplication, division, etc.



(a) Block Diagram



MAIN REG SET		ALTERNATE REG SET		
ACCUMULATER A	FLAGS F	ACCUMULATOR A'	FLAGS F'	
B	C	B'	C'	GENERAL PURPOSE REGISTERS
D	E	D'	E'	
H	L	H'	L'	

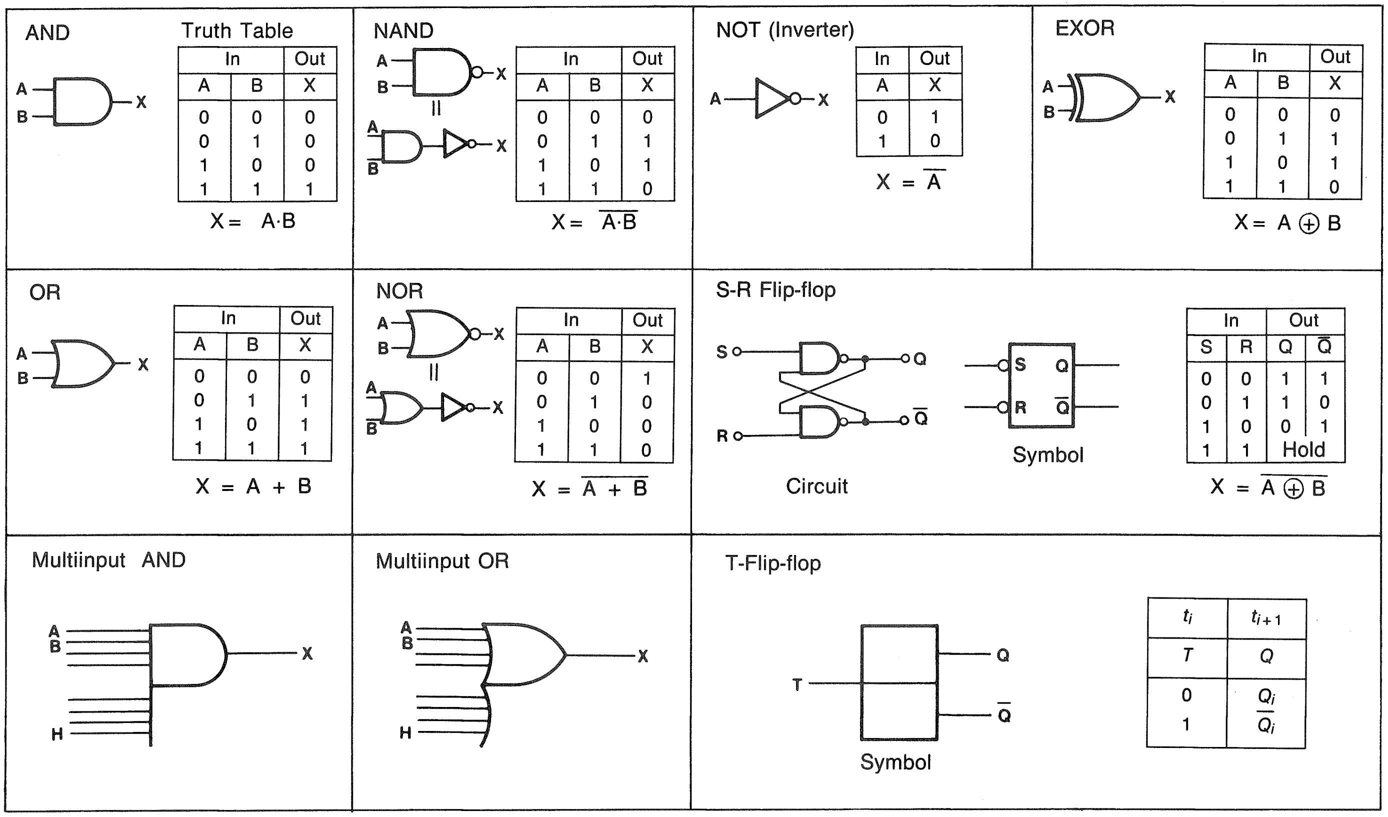
INTERRUPT VECTOR I	MEMORY REFRESH R	SPECIAL PURPOSE REGISTERS
INDEX REGISTER IX		
INDEX REGISTER IY		
STACK POINTER SP		
PROGRAM COUNTER PC		

(b) Register Array

Fig. 8 Z80A CPU

Logic devices

Fig. 9 shows the basic AND, NAND, OR, NOR, NOT EOR, and flip-flop circuits. Flip-flops are widely used as register and memory devices.



AND

In		Out
A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

$X = A \cdot B$

NAND

In		Out
A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

$X = \overline{A \cdot B}$

NOT (Inverter)

In	Out
A	X
0	1
1	0

$X = \overline{A}$

EXOR

In		Out
A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

$X = A \oplus B$

OR

In		Out
A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

$X = A + B$

NOR

In		Out
A	B	X
0	0	1
0	1	0
1	0	0
1	1	0

$X = \overline{A + B}$

S-R Flip-flop

In		Out	
S	R	Q	$\overline{Q}$
0	0	1	1
0	1	1	0
1	0	0	1
1	1	Hold	

$X = \overline{A \oplus B}$

T-Flip-flop

t_i	t_{i+1}
T	Q
0	Q_i
1	$\overline{Q_i}$

Fig. 9 Logic Devices

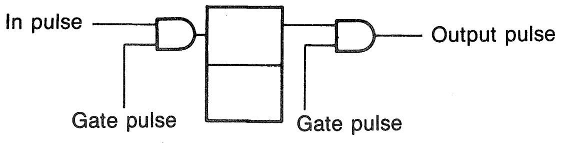
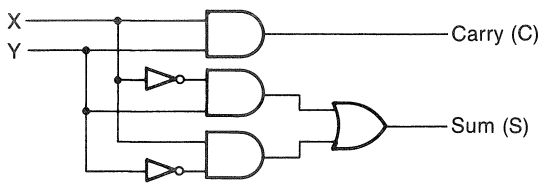


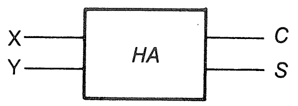
Fig. 10 AND Gate

Half adder

This accepts two binary input signals and produces corresponding sum and carry outputs and has no carry input from the previous stage.



(a) Logic Circuit



(b) Symbol

Add		Result	
X	Y	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

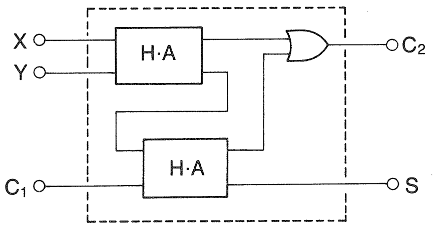
$S = X \cdot \overline{Y} + \overline{X} \cdot Y$
$C = X \cdot Y$

(c) Truth Table

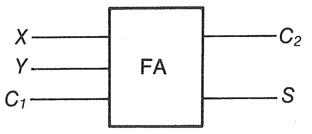
Fig. 11 Half Adder

Full adder

This accepts two binary digits to be added and a carry digit from the previous stage and produces sum and carry outputs.



(a) Circuit



(b) Symbol

Add			Result	
X	Y	C_1	S	C_2
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$S = X \cdot Y \cdot C_1 + \overline{X} \cdot Y \cdot \overline{C_1} + \overline{X} \cdot \overline{Y} \cdot C_1 + X \cdot \overline{Y} \cdot \overline{C_1}$$
$$C_2 = X \cdot Y + X \cdot C_1 + Y \cdot C_1$$

(c) Truth Table

Fig. 12 Full Adder

Fig. 13 shows a 4-bit parallel adder. If 0110 and 1101 are added, inputs and outputs are as follows:

Bits	Carry	4	3	2	1
Inputs A		0	1	1	0
Inputs B		1	1	0	1
Inputs C	0				
Outputs S	1	0	0	1	1

Carry

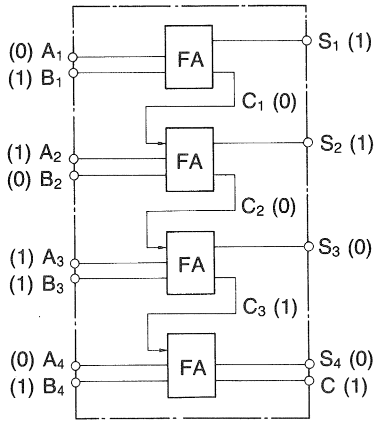


Fig. 13 4-bit Parallel Adder

Accumulator (A-register)

A register for temporarily storing a number before and after arithmetic, logical or transferral operation.

Temporary Register (B-register)

This is the subsidiary register for Accumulator, holds one of the arithmetic operands and delivers it to the ALU.

Instruction Register (Control register)

This retains the code of the instruction currently being executed.

Flag Register

7	6	5	4	3	2	1	0
S	Z		H		P/V	N	C

The result of operation or the current conditions of the computer are stored here. Referring to the conditions the computer determines what to do next. For example, you can jump and stop the operation by arranging the program:

If EVEN,	JUMP to address 8500
If NEGAtive,	JUMP to address 8600
If ZERO,	STOP

S: Sign flag. This bit becomes "1" if the result of arithmetic operation is negative.

Z: Zero flag. "1" if the result of the operation is zero.

H: Half-carry flag. "1" if the add or subtract operation produced a carry into or borrow from bit-4 of the accumulator.
Refer to DAA, page 56.

P/V: Parity or overflow flag . Parity and overflow share the same flag. Logical operations affect this flag with the parity of the result while arithmetic operations affect this flag with overflow of the result. If P/V holds parity and if the result of the operation is even, "1". If P/V holds overflow and the result of the operation produced an overflow, "1".

N: Add/Subtract flag. "1" if the previous operation was a subtract.
H and N are used in conjunction with the decimal adjust instruction (DAA) to correct the result in BCD arithmetic. The knowledge of these functions of this register is required for making a program although the contents of this register is not directly accesible to programmers.

C: Carry flag. "1" if the operation produced a carry from the MSB of the operand or result.

Bits 3&5: Not used

General-purpose Registers

Internal addressable registers in a CPU that can be used for temporary storage, as accumulators, index registers, stack pointers, or for any other general-purpose functions.

Special Registers

Stack pointer: When a program is sustained, the interim result should be stored somewhere in the memory to be used later. This enables the CPU to process "stack" of data using the last-in first-out (LIFO) method. This contains the address number of the current top or entry point of the stack of information. When an interrupting service is finished, the stacked information is popped or taken out for resuming sustained program. This will be discussed later.

Program Counter: This holds the address value of the memory location where the next CPU directive is to be obtained.

Index Register: This contains a quantity to be added to the address number of the instruction under the direction of the program for automatic and temporary address modification.

Control unit

This controls the operation of the whole computer system by directing the sequence of operation, decodes instructions and gives commands particular devices to start or stop.

Clock

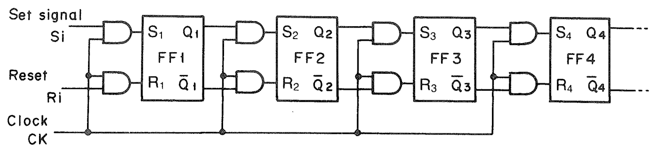
The operation of the most microcomputers is synchronized with a clock signal. The operating speed of a computer depends on the clock frequency. Z80A employs a 4.0MHz external clock.

Buffers

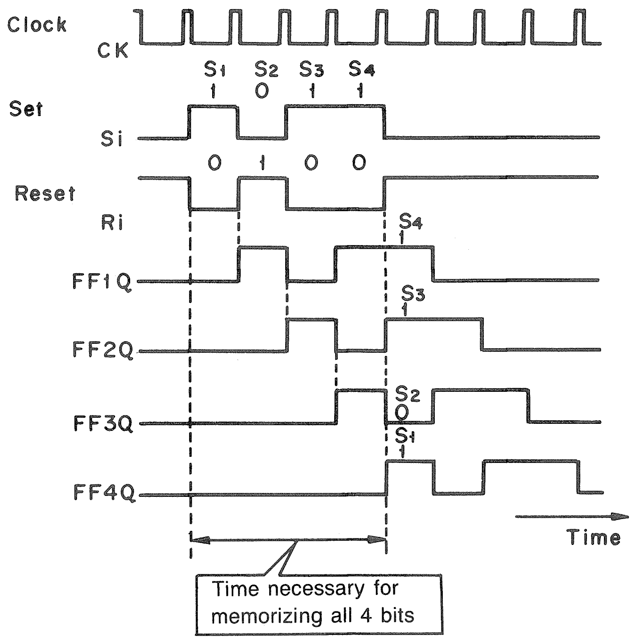
In addition to impedance matching devices, these work as storages used to compensate for a difference in rate of flow of data, or time of occurrence of events when transmitting data from one device to another.

Shift register

The following is a right-shift register. The inputs S_i and R_i have a correlation of $S_i \neq R_i$ and appear in series synchronously to the clock signal. Q and $\overline{Q}$ appear 1-clock later than the time when the inputs appear. They are shifted to the right one by a clock pulse. This is also used as a serial-parallel converter and a shifter for arithmetic and logic operation.



(a) Circuit

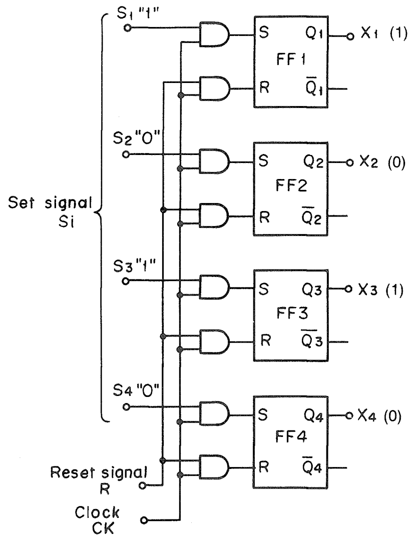


(b) Timing chart

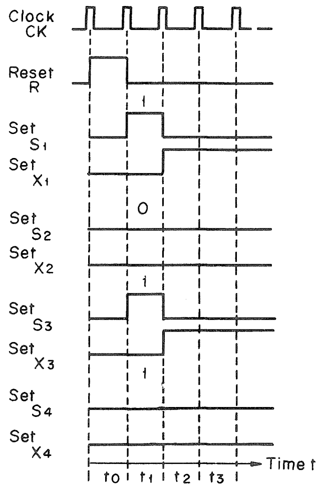
Fig. 14 Shift Register

Parallel register

For storing information, this register is first cleared by Reset signal during t_0, then Set signal is applied to the S-inputs and only the flip-flops applied with "1" shift their outputs during t_1. The outputs are kept unchanged till the next Reset signal is applied.



(a) Circuit

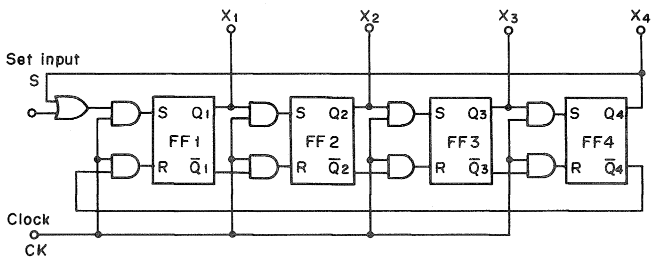


(b) Timing Chart

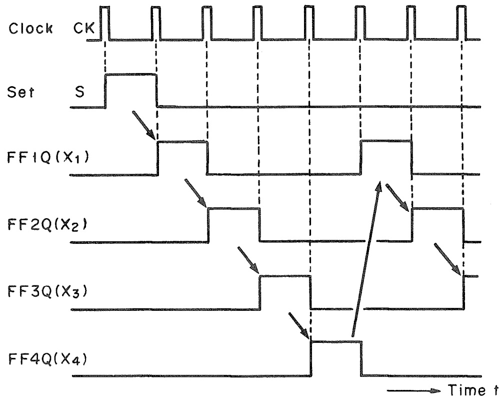
Fig. 15 Parallel Register

Ring counter

This is a kind of shift registers, carries only only one positive pulse at a time and is used for controlling the timing for reading and executing instructions.



(a) Circuit

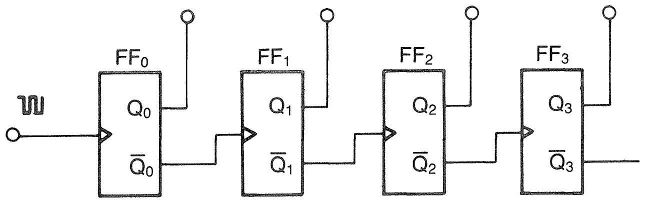


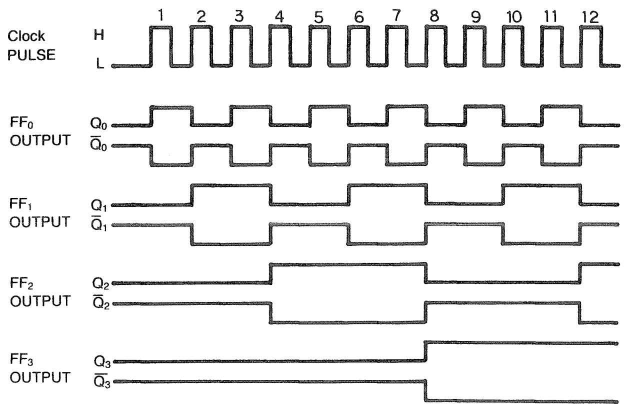
(b) Timing Chart

Fig. 16-1 Ring Counter

Binary counter

This produces one output pulse for every two input pulses and is used for the program counter.





(b) Output waveform

Fig. 16-2 Binary Counter

Decoder

This selects one or more output channels according to the combination of input signals, determines the meaning of an instruction from the combination and initiates a computer operation. If the instruction code is made of 8 bits, 256 kinds (2^8) of outputs are available. Fig. 17 shows the way an operation code is fetched and decoded.

In the Z80A the instruction code "10000000" means that "Add the contents of A and B registers and place the result in the A register, or "(A) ← (A) + (B)".

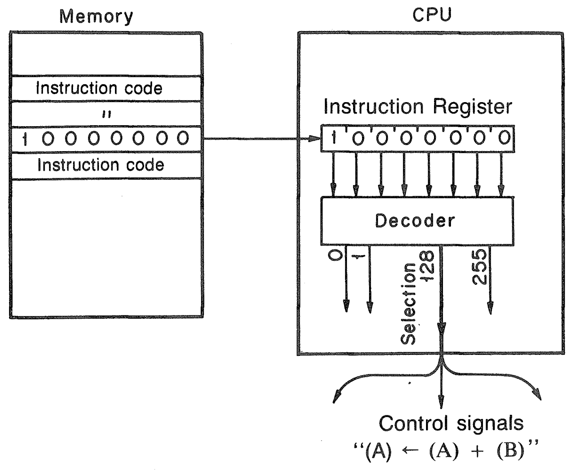
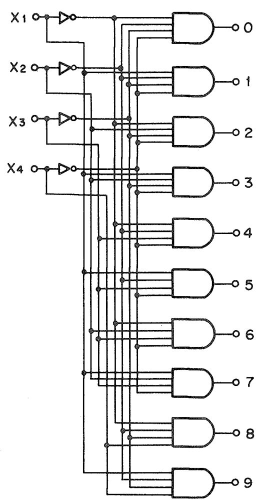


Fig. 17 Instruction Decoding ↗

If 10000000 (B) is applied from the Instruction Register to Decoder, 128th output is selected. The signal for performing the addition is applied to the devices concerned in CPU, and the instruction is executed. Fig. 18 shows a binary-decimal decoder.



Truth Table

In				Out
X_4	X_3	X_2	X_1	Y
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	1	8
1	0	0	1	9

Fig. 18 Binary Decimal Decoder

2.4 Interface

I/O Interface

The CPU understands only its own machine language made of binary codes while peripheral equipment deal with various types of signals. The interface is a common boundary between the CPU and peripheral equipment and adapts the signal requirements of the equipment to those of a computer and vice versa. Its functions are:

a. Adjusting signal format, mode, characteristics and speed (Digital ↔ Analog, level, current, speed, etc.)
b. Selecting one peripheral equipment from those connected
c. Decoding a CPU's operation command and passing it to the equipment concerned
d. Memorizing data temporarily
e. Informing CPU of the conditions of the equipment or vice versa

Many kinds of interfaces such as serial input-output interface (SIO) and parallel input-output interface (PIO) are required to deal with various peripheral equipments. PX-7 equips interfaces for LaserVision players and CRT display in addition to the MSX standard interface for keyboard, cassette recorder, TV monitor, joystick, tablet, etc. SIO arranges parallel data to serial or vice versa. Buffers adjust signal level, current and the timing of data transmission holding received data temporarily.

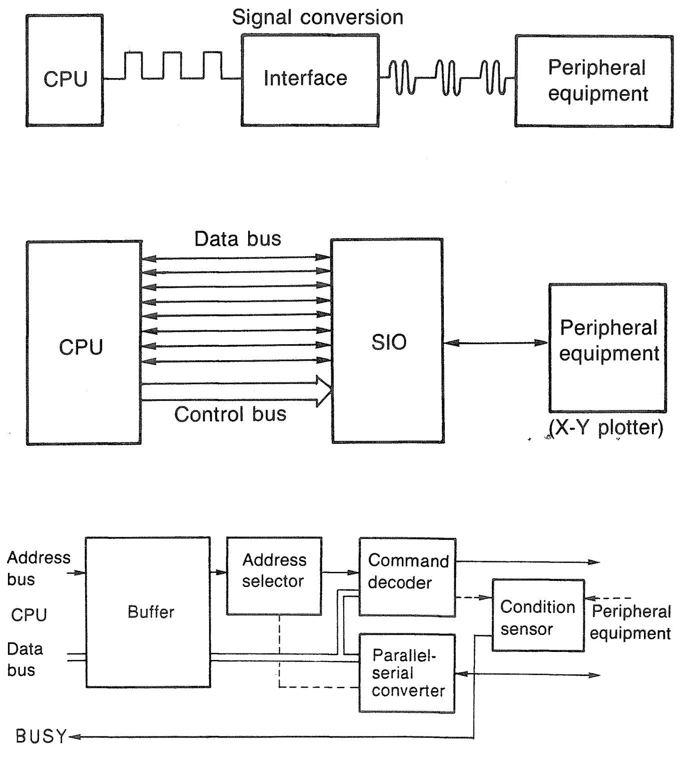
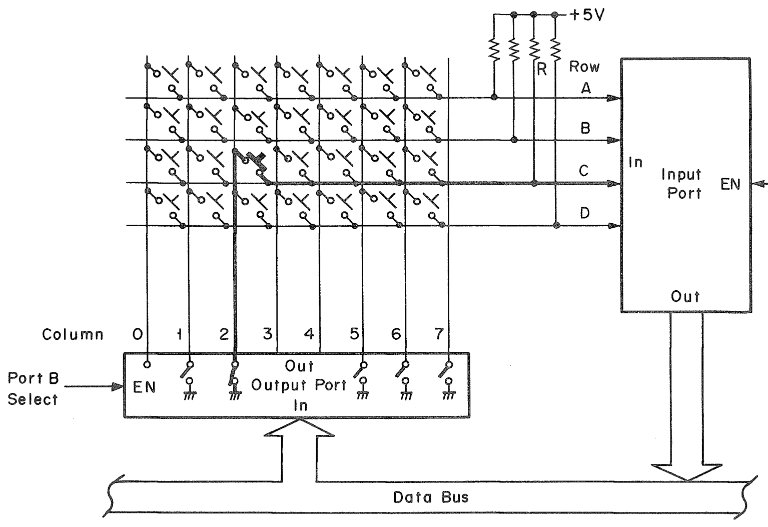


Fig. 19 Interfacing (an example)

Keyboard

Let's see the basic operation of a keyboard. Fig. 19 shows the interface of the 32-key board. A program is required to scan the keyboard quickly. The output port scans the columns to set one of them low and the input port reads the rows in turn. If the key 2-C is pressed and the line 2-C is closed, the current flows from +5V to Column 2 through the R and the key swicth 2-C, and the level at C is pulled down when the Column 2 is set low.



Then the input port can detect that the switch 2-C is being depressed.

Fig. 20 Keyboard

Character Display

In Fig. 21 the character blocks are quickly switched on and off in turn and only one of them is on at any moment. The outputs of the segment driver are changed synchronizing with the scan timing of the block driver. So the character information for the block 1 appears only when the block 1 is turned on.

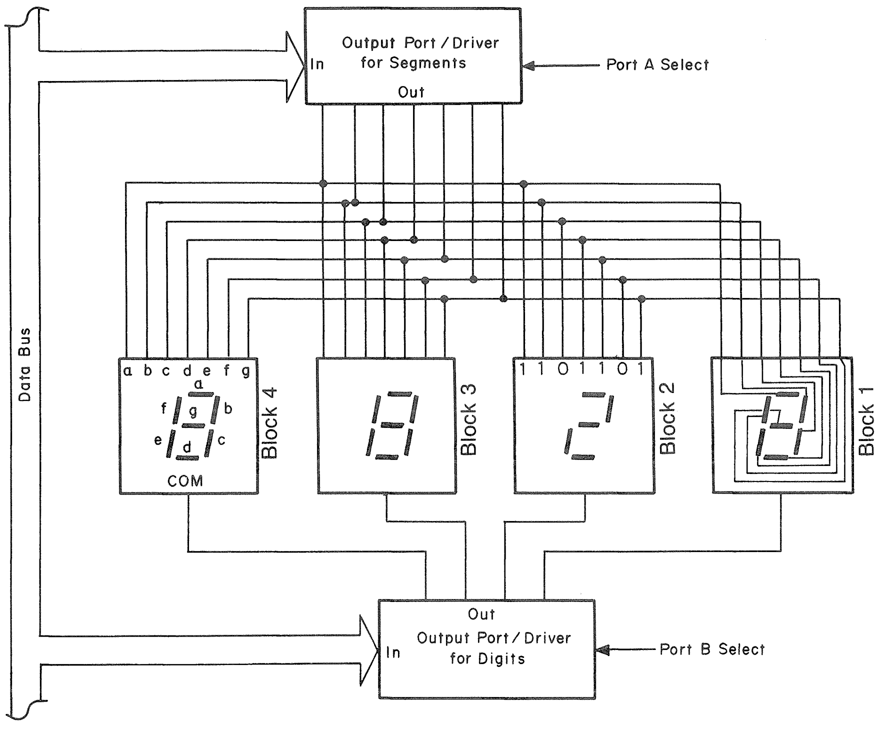


Fig. 21 Character Display

2.5 Bus Structure

The bus is a set of wires over which information is transferred. Basic computer signals are address, data and control signals. A computer has an address bus, I/O port bus, data bus and control bus to let its devices exchange data with the processor. In the Z80A system I/Os and memories share the same bus.

Address bus: A unidirectional bus over which appears digital information that identifies either a particular memory location or I/O device. No matter how many memory locations there may be, only one may be accessed at a time. The number of address lines of Z80A CPU is 16. 16-line address bus can address 65536 locations (2^{16}) directly that can keep 65536-byte information because one memory location keeps 8-bit (or 1-byte) information.

Data bus: Bidirectional data lines. Z-80A has 8 data lines for sending 8-bit information in parallel at a time. Larger computers have separate input and output data buses. To save the bus lines, microcomputers use the 8 lines for both input and output purposes by sharing time. To commonly use the bus, a traffic controller is required.

Control bus: A set of control lines, usually from 10 to 100, with a function to carry the synchronization and control information necessary to the computer system. Z80A has 13 control lines, six lines for system control, five for CPU control and two for CPU bus control. The bus carries a signal to indicate whether the current transaction is a read or write from a memory or a peripheral, interrupt request or bus request signal, an extended cycle signal, and a response to indicate that the request of a peripheral device has been accepted and so on. Such signals might be interrupt, hold, acknowledge, read, write, etc.

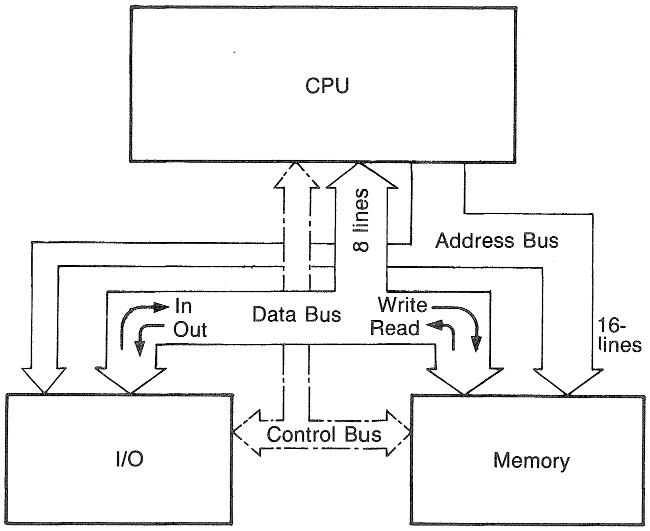


Fig. 22 Bus Structure

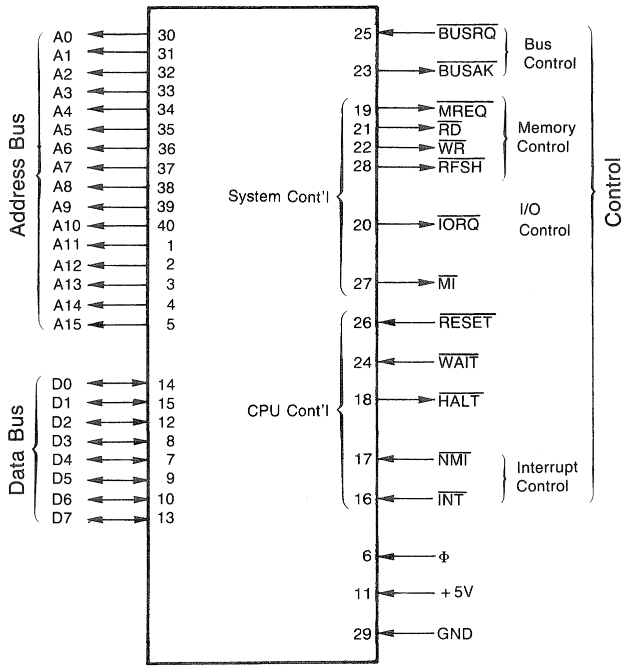
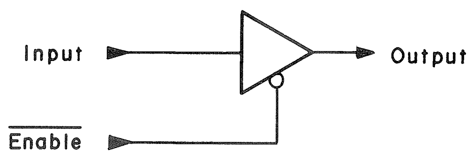


Fig. 23 Terminals and Bus Lines of Z80A

3-state Bus

The microprocessor selects one device to talk to and disconnects the others with the help of the tristate drivers (or buffers) that enable the processor to selectively turn devices on and off. In Fig. 24, only when the Enable is low (L), the buffer passes the input signal to the output.



Enable	Input	Output
0	0	0
0	1	1
1	0	floating
1	1	floating

0 = low, 1 = high, floating = high impendance

Fig. 24 Tristate Driver

A bidirectional buffer can be formed with two tristate buffers and an inverter. The control signal determines the direction of signal flow.

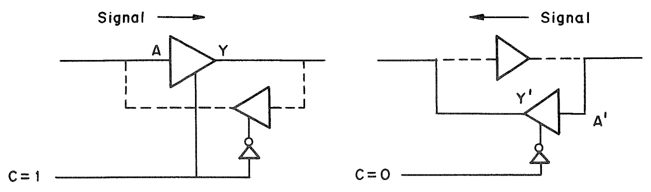
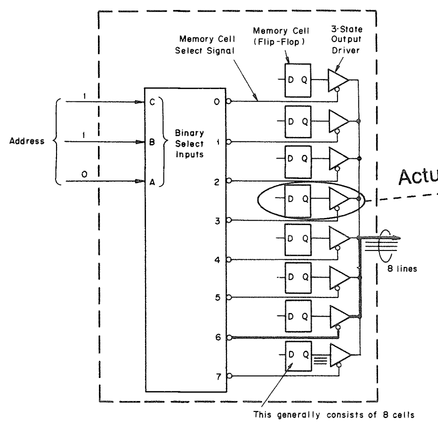


Fig. 25 Bidirectional Buffer ↗

Fig. 27 shows how a memory cell is selected sharing one signal data line. When address number is 110 (B), the cell No. 6 (D) is selected and its data is put out on the data line.



Memory Cell Select Signal
Memory Cell (Flip-Flop)
3-State Output Driver
Address
1
1
0
C
B
A
Binary Select Inputs
0
1
2
3
4
5
6
7
D Q
8 lines
This generally consists of 8 cells
Actually 8 Flip-flops similar to this.

Fig. 27 Selecting A Memory Cell

In Fig. 26 four devices share only one data line in communication.

Actually there are more senders and receivers. Each sender/receiver is provided with two control signals: Output Enable for its tristate driver and Data Strobe for input. The control logic permits only one sender to be active at a time. Other disabled drivers are in a high-impedance (floating) state and put out neither H nor L signal, and they have no effect on the logic state of the bus. When Device-A sends a piece of data to Device-2, the Control sets Output-enable-A true and all the other Output-enables false, and then sends a pulse on the line of Data-strobe-2 to let the Device-2 receive the data. RAMs are typical senders/receivers. This method saves connecting lines.

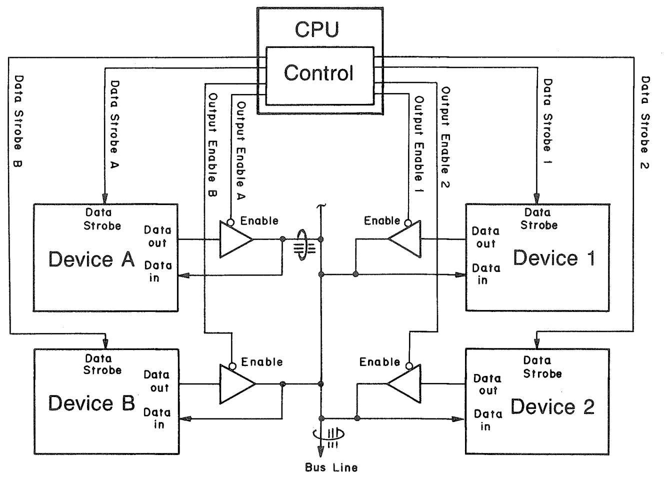


Fig. 26 Tristate Bus In Communicating System

In Fig. 28 the eight flipflops are enabled at a time, and put out parallel 8-bit data.

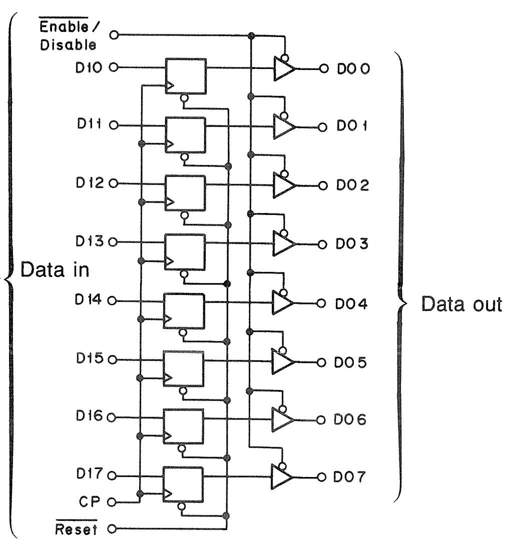
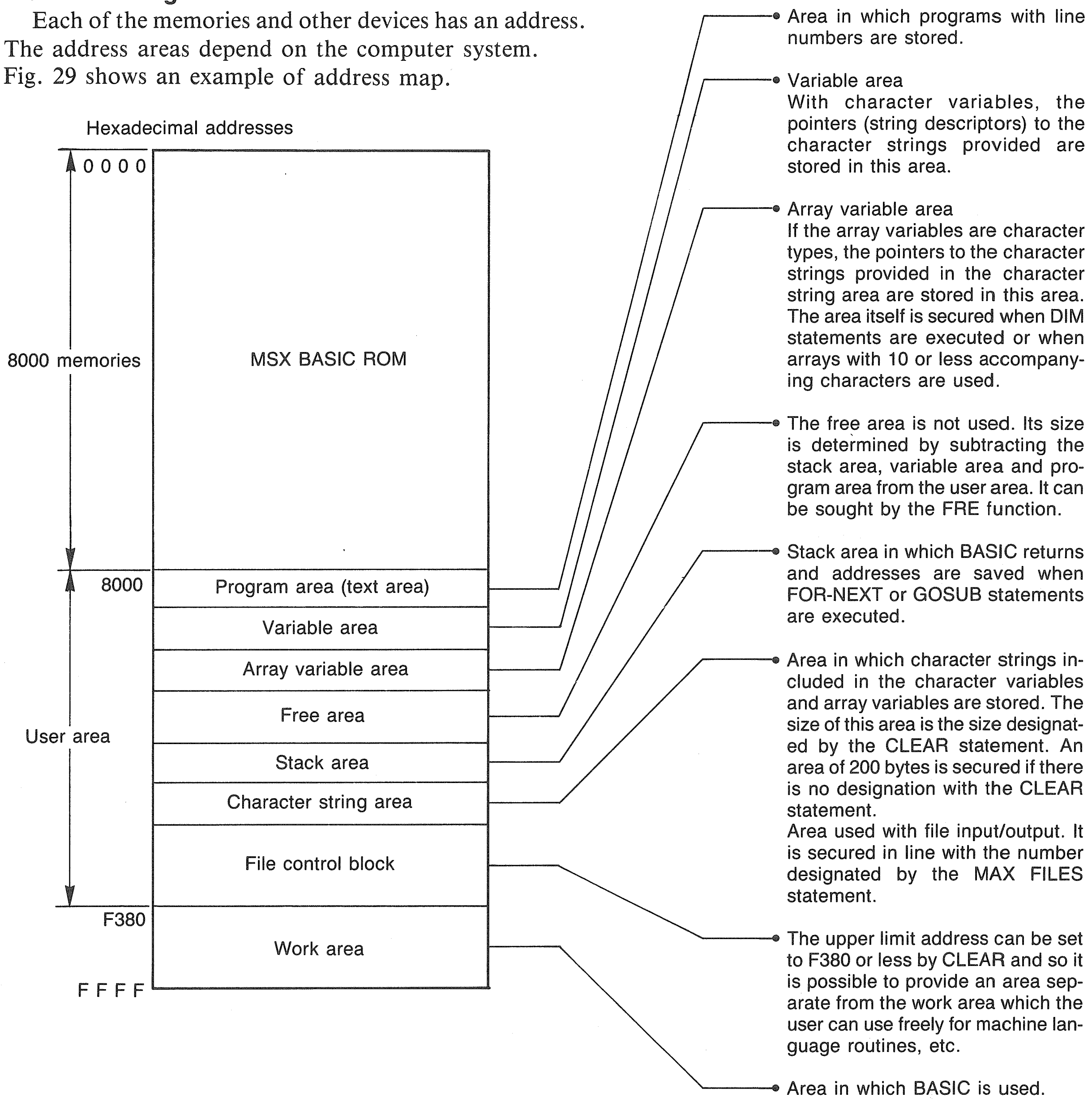


Fig. 28 8-bit Register with Tristate Gates

2.6 Addressing structures

Each of the memories and other devices has an address. The address areas depend on the computer system. Fig. 29 shows an example of address map.



- Area in which programs with line numbers are stored.
- Variable area
  With character variables, the pointers (string descriptors) to the character strings provided are stored in this area.
- Array variable area
  If the array variables are character types, the pointers to the character strings provided in the character string area are stored in this area. The area itself is secured when DIM statements are executed or when arrays with 10 or less accompanying characters are used.
- The free area is not used. Its size is determined by subtracting the stack area, variable area and program area from the user area. It can be sought by the FRE function.
- Stack area in which BASIC returns and addresses are saved when FOR-NEXT or GOSUB statements are executed.
- Area in which character strings included in the character variables and array variables are stored. The size of this area is the size designated by the CLEAR statement. An area of 200 bytes is secured if there is no designation with the CLEAR statement.
  Area used with file input/output. It is secured in line with the number designated by the MAX FILES statement.
- The upper limit address can be set to F380 or less by CLEAR and so it is possible to provide an area separate from the work area which the user can use freely for machine language routines, etc.
- Area in which BASIC is used.

Fig. 29 Memory Map of PX-7

Selecting Memory Location

Fig. 30 shows a 2K × 8 RAM. 2k RAM has 2,048 (2^{11}) address locations and 16,384 bits (2,048 bytes × 8 bits). Each byte has its own address. The RAM has 10 address inputs, 8 data in/out terminals and one each of chip select and Read/Write (R/W) terminals. ROMs have no R/W terminal and the data bus is unidirectional because they can not be written. When writing, the Chip Select signal becomes true, address signal appears on the address bus, a particular memory location is waken up, data are placed on the data bus, and then the R/W signal is made low.

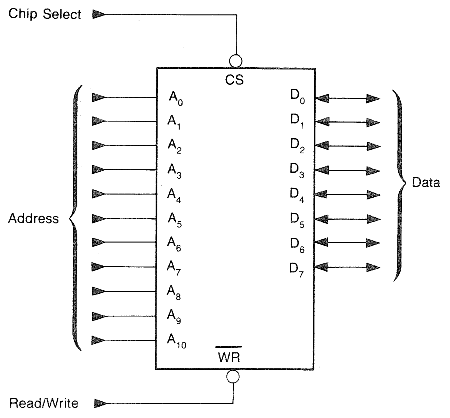


Fig. 30 2K x 8 RAM

Address Decoding

A microcomputer uses many memory chips and some serial addresses are assigned to each chip. If the chip has 256 locations, the number of the higher order byte of addresses will be the same. The higher order half of the address bus can be saved by decoding the 9th (A8) and 10th (A9) bits preliminary and selecting a memory chip. Fig. 31 shows that 256 addresses are assigned to each of three 256 byte ROMs. In Fig. 32, the address lines from A10 to A15 can be ANDed because the signal on the lines is always 0. A8 and A9 determine the chip to be selected. The information on the lower half of the address bus is applied to all chips, but only one selected chip reads the address at a time because the others are all disabled. 8 lines from A0 to A7 determine the memory location to be selected inside the memory chip.

		ROM NUMBER								LOCATION WITHIN ROM							
	Address \ Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ROM 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
	⋮																
	255	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
ROM 1	256	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
	257	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
	⋮																
	511	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
ROM 2	512	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
	513	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
	⋮																
	766	0	0	0	0	0	0	1	0	1	1	1	1	1	1	1	1
		Always "0"						Select a chip		Select a memory location inside a chip							

Fig. 31 Addresses Assigned to Each of Three 256-Byte ROMs in a System

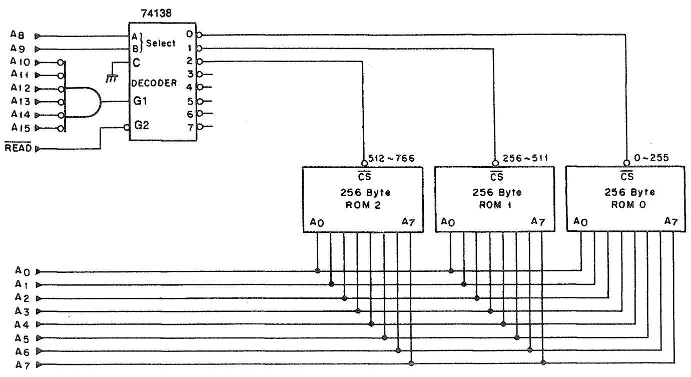


Fig. 32 Selecting Three 256-Byte ROMs

Fig. 33 is the block diagram of a RAM circuit. The address lines are connected in matrix as shown in Fig. 5 and 6.

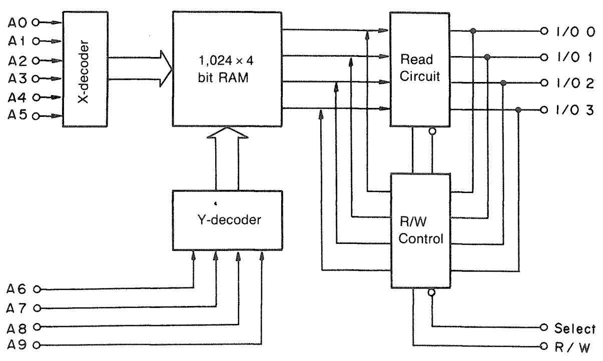


Fig. 33 Inside a Static RAM

3. Control signals

Various control signals are provided to control the system and the flow of information. These signals travel over the control lines and do not transmit data but determine what should be done next. The CPU first receives an instruction from a memory or external device and stores it in Instruction Register, decodes it in Instruction Decoder, and then generated necessary control signals.

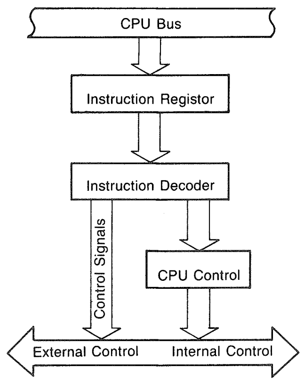


Fig. 34 Decoder & Control ↗

There are two kinds in Control signals.
One for controlling inside the CPU and the other is for controlling outside. All operations of microcomputer are synchronized with the clock pulses generated inside the computers.
Fig. 33 shows the control signals of Z80A CPU.

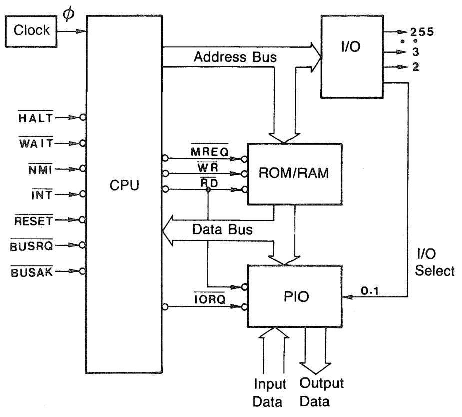


Fig. 35 Z80A's Control Signals

3.1 Memory and I/O Controls

$\overline{\text{MREQ}}$ (Memory Request) signal

There are two address signals, memory address signals up to 65536 and I/O address signals up to 256. The $\overline{\text{MREQ}}$ signal is to announce that a memory address signal is on the address bus.

$\overline{\text{IORQ}}$ (I/O request)

This is to announce that an I/O address signal is on the address bus. I/O address signals require the low-order eight lines of the address bus because CPU addresses up to 256 locations.

$\overline{\text{WR}}$ (write)

With this signal CPU informs memory and peripheral input/output that it has put out data onto the data bus. In writing, address signal is placed on Address bus, $\overline{\text{MREQ}}$ (or $\overline{\text{IOREQ}}$) signal shifts to L, data are placed on the data bus and then $\overline{\text{WR}}$ signal is put out on $\overline{\text{WR}}$ line.

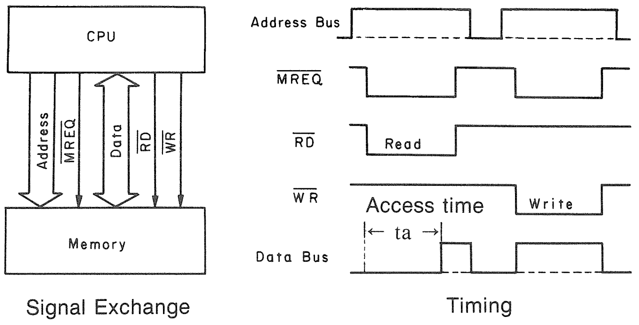


Fig. 36 CPU Puts Out $\overline{\text{MREQ}}$ Signal When Reading From or Writing to a Memory.

$\overline{\text{RD}}$ (Read)

With this signal CPU informs memory and IOs that it is going to take data in. The addressed memory or IO can put out data when $\overline{\text{MREQ}}$ (or $\overline{\text{IOREQ}}$) and $\overline{\text{RD}}$ shift to L. $\overline{\text{RD}}$ signal stays low longer than $\overline{\text{WR}}$ signal because it takes some time for memories and IOs to put data out. The time is called "access time". It may be 10ns — 100ns. If it requires longer, $\overline{\text{WAIT}}$ signal can be used.

$\overline{\text{RFSH}}$ (Refresh)

Dynamic RAMs can keep information for a very short time and require refreshing before it is cleared. Refresh pulse cause rewriting the same information

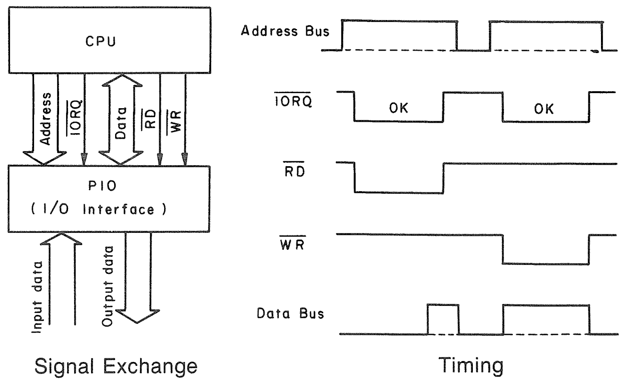


Fig. 37 CPU Puts Out $\overline{\text{IORQ}}$ Signal When Reading From or Writing to an I/O Port.

3.2 Bus controls

These signals control data and address buses. Data bus is in one of the three states; read, write and floating. Address bus is in either of the two state; the state it carries an address signal and that it does not carry the signal.

DMA (direct memory access)

When transferring data by a program, they are once taken in and then put out by CPU that takes some time although it is very short. DMA is a technique that allows a computer user to have direct access to individual memory locations bypassing CPU with the help of DMA controller.

$\overline{\text{BUSRQ}}$, $\overline{\text{BUSAK}}$ (Bus control signal)

In such a case DMA controller sends $\overline{\text{BUSRQ}}$ (bus request) signal to CPU, when CPU receives the signal, it makes address and data buses into high impedance state and sends the controller $\overline{\text{BUSAK}}$ (bus acknowledged) signal. In the high impedance state, the address and data buses are not affected by the CPU. Then the controller starts controlling the DMA operation. Prior to this transfer CPU should provide the controller with the destination address and quantity of data to be transferred.

Cycle stealing

The more efficient way of using a computer is cycle stealing. With this method, DMA becomes possible while CPU is doing internal work and not using the buses. DMA is suspended while CPU is using the buses.

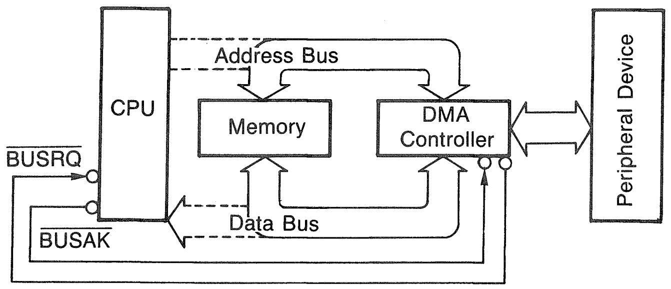


Fig. 38 Direct Memory Access

3.3 Other controls

$\overline{\text{INT}}$ (Interrupt)

This signal, usually caused by a signal from an external source, requests the CPU a break in the normal flow of a program to insert other job of higher priority. When interrupt is acknowledged, the CPU diverts its attention from the address of the main program currently communicating to the new address directly related to the type of interrupt that has occured. The flow can be resumed from the suspended point when the interrupt service is completed.

$\overline{\text{NMI}}$ (Nonmaskable Interrupt)

This is an interrupt signal not maskable or ignorable by program instructions.

$\overline{\text{HALT}}$

This is to announce that CPU is in stop status. CPU can be stopped by a $\overline{\text{HALT}}$ instruction inserted in the program. It can be restarted with a Reset signal or interrupt signal.

$\overline{\text{WAIT}}$

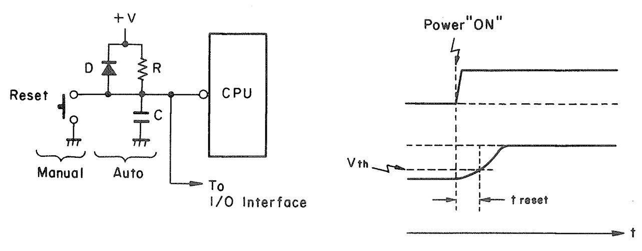
CPU works very fast compared to peripheral equipment especially when writing to the equipment. In such a case the CPU can be kept waiting by applying it to $\overline{\text{WAIT}}$ terminal with L singal to keep pace with the peripheral equipment.

$\overline{\text{RESET}}$

When $\overline{\text{RESET}}$ level is made L, CPU and I/O interfaces are brought into the initial state. While it is L, the address bus and data bus go into high impedance state and control signals do not affect them.
When Power is turned on, the $\overline{\text{RESET}}$ terminal is kept low for a short time by an RC circuit and the Program Counter (PC) is cleared automatically. This circuit is popularly employed in audio and video equipment.

$\overline{\text{M1}}$ (Machine Cycle 1)

Low $\overline{\text{M1}}$ signal indicates that CPU is in the OP code fetch cycle. (Discussed later)



(a) Manual Reset and Auto Reset (b) Auto Reset Timing

Fig. 39 Reset Circuit

4. Timing

The operation of CPU is the repetition of instruction fetch and execution.

The time interval required for an element of operation is called a "state", that for executing one operation is called a "machine cycle" and that for completing one instruction is called an "instruction cycle". The first machine cycle is the operation-code-fetch cycle with the length of 4 — 5 states. In this cycle the operation code (OP code) is fetched. In the next machine cycle (3 — 4 states) data are exchanged between the CPU and a memory or I/O device. The length of the machine cycle can be extended by the $\overline{\text{WAIT}}$ signal for synchronizing the operation of CPU with that of the communicating external device. One state or T cycle will take 0.25μs if the clock frequency is 4MHz. Then the time necessary for executing a program can be known by multiplying the total states by 0.25μs.

Byte	7	6	5	4	3	2	1	0	
B1	0	1	0	0	1	1	1	1	Instruction fetch
									Execute
B1	0	0	1	1	0	0	1	0	Instruction fetch
B2	0	0	0	0	0	0	0	0	
B3	0	1	0	0	0	0	0	0	
									Execute
B1	1	1	0	0	0	1	1	0	Instruction fetch
B2	1	1	1	1	1	1	1	1	
	………………………………								Execute

Fig. 40 Sequential Operation of CPU

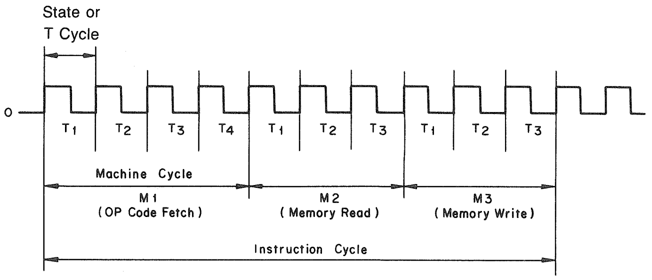


Fig. 41 Typical Timing of CPU

4.1 OP-code fetch

Fig. 42 shows the timing diagram of OP-code fetch cycle. In the state T_1, the content of PC is placed on the address bus. $\overline{\text{MREQ}}$ signal, meaning address is valid, becomes active. $\overline{\text{RD}}$ signal becomes active to make the system ready to put the data on data bus. Then data is placed on the data bus. The CPU reads the data at the rising edge of the T_3 clock signal ϕ and makes the $\overline{\text{MREQ}}$ and $\overline{\text{RD}}$ signals inactive. In T_3 and T_4, OP code is decoded. At the same time $\overline{\text{RFSH}}$ shifts to L, then $\overline{\text{MREQ}}$ shifts to L, and dynamic memory is refreshed by the two signals. When the $\overline{\text{WAIT}}$ signal becomes active, the CPU samples it at the falling edge of ϕ at T_2 and waits for 1T.

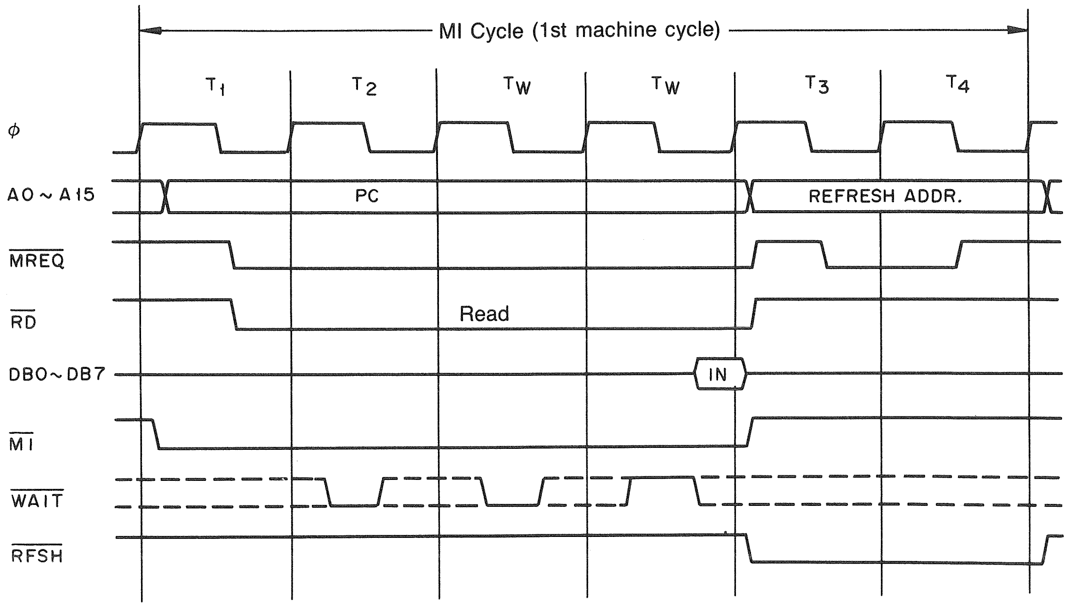


Fig. 42 OP Code Fetch Cycle with Two Wait States

4.2 Memory read and write

One memory read or write cycle is 3-state long unless it is extended by $\overline{\text{WAIT}}$ signal. $\overline{\text{MREQ}}$ and $\overline{\text{RD}}$ work the same as in the fetch cycle. $\overline{\text{MREQ}}$ becomes active after address bus becomes stable, and the content of data bus disappears after $\overline{\text{WR}}$ becomes inactive.

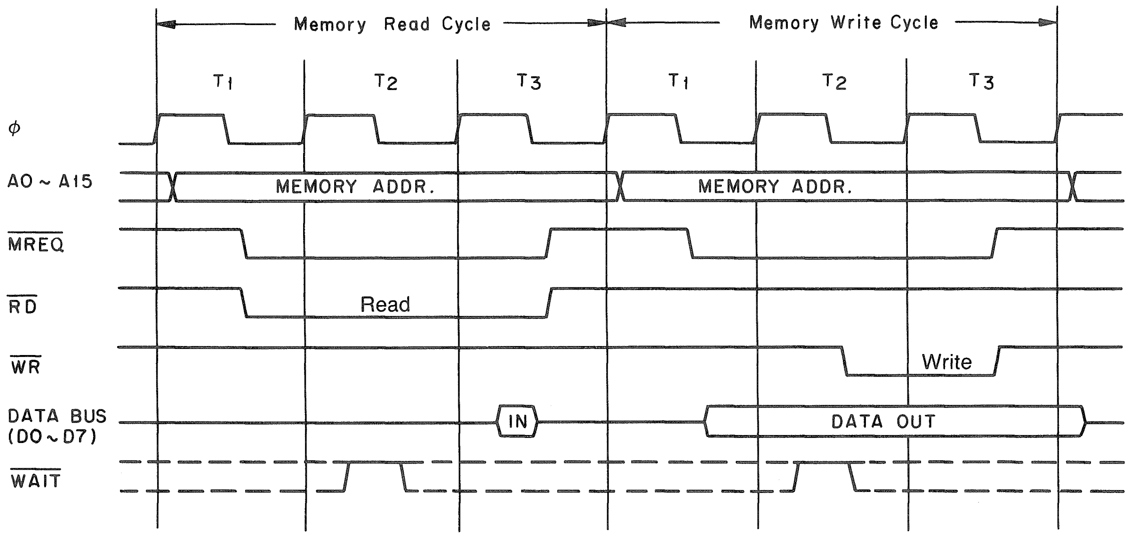


Fig. 43 Memory Read and Write Cycles

4.3 Interrupt/Acknowledge cycle

The CPU samples the $\overline{\text{INT}}$errupt signal at the rising edge of the clock signal in the last state of an instruction cycle if the interrupt enable flip-flop is set and $\overline{\text{BUSRQ}}$ (bus request for DMA) is not active. Then, the $\overline{\text{M1}}$ becomes active and M1 cycle peculiar to the interrupt starts, and PC changes its content. $\overline{\text{IORQ}}$ becomes active and allows the interrupting device to place data on the data bus. Here two waiting states are inserted into this cycle automatically to wait until the ripple is smoothed and to verify what I/O device is requesting the interrupt. The data is fetched on the rising edge of the T3 clock.

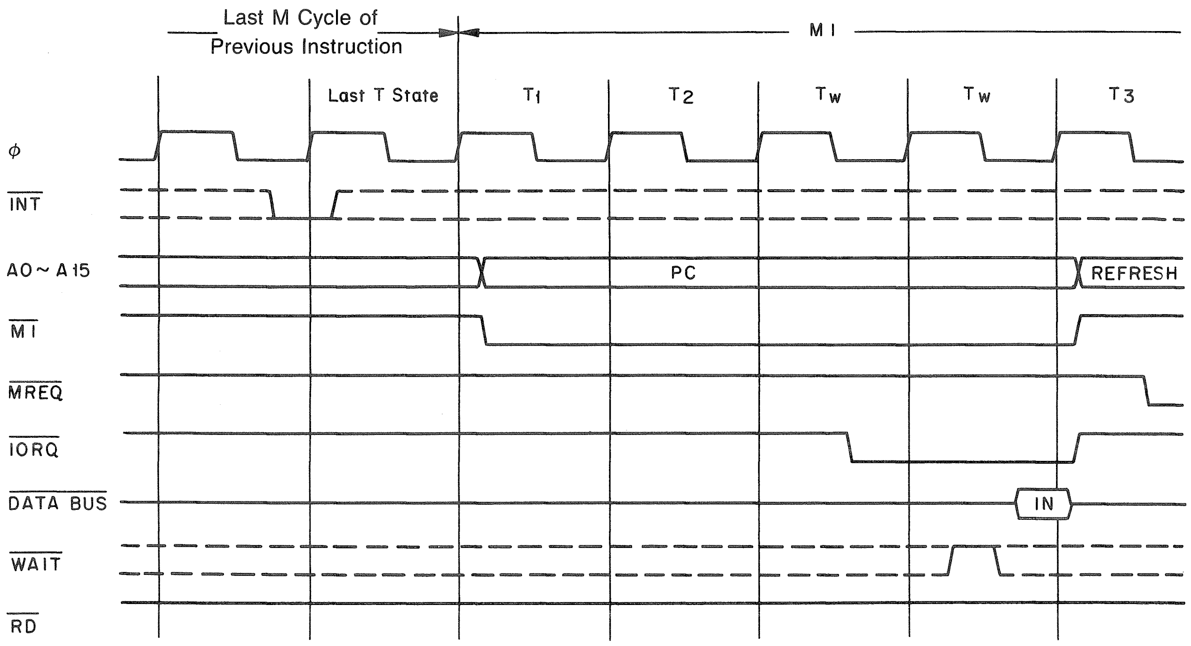


Fig. 44 Interrupt Request/ Acknowledge Cycle

5. Computer Languages

Each computer has its own language called machine language. The language is hard to understand because it is written with binary codes. Although hexadecimal representation is a little easier to understand than the binary representation, it still does not make sense to us. Mnemonic codes assigned to decimal numbers, characters, symbols and instructions are much better. For easier understanding high level languages such as FORTRAN, COBOL and BASIC have been developed. Here we will discuss some words of ASSEMBLY language and machine language to see the operation inside the computer because the ASSEMBLY language is the closest to the machine language.

Instructions

Usually a set of instruction consists of an instruction itself and data to be processed or an address to call where the referenced operand is to be stored.

Each microprocessor has its own machine language. A statement of the assembly language consists of three parts; Label, Mnemonic and Operand. They are translated into the machine language.

Assembly word

Label	Mnemonic	Operand	Comments
FILE:	LD	B,E	; Move register E to register B ((B) ← (E))
Machine word	OP code	Address or data	1 byte, 8 bits

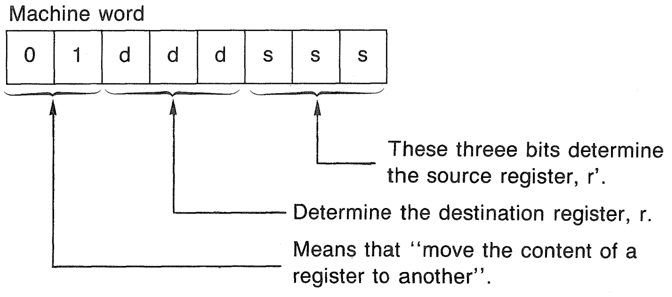
Label is a set of symbols for identifying a line of a program when you want to refer to it in the following instructions. (:) is for identifying the label. The Comments describe the meaning of the Assembly words. (;) indicates the beginning of a comment. One instruction consists of one or a few bytes.
The instructions can be classified as follows:

a. Data transfer instructions
b. Arithmetic and logical instructions
c. Branch instructions
d. Stack, Input/output instructions
e. Control instructions, etc.

5.1 (a) Transfer instructions

Transfer between registers

(1) Instruction: **LD r, r'**
Comment: Move register to register
(r) ←(r')
r: One of the CPU registers A, B, C, D, E, H, L.


Machine word
0
1
d
d
d
s
s
s
These threee bits determine the source register, r'.
Determine the destination register, r.
Means that "move the content of a register to another".

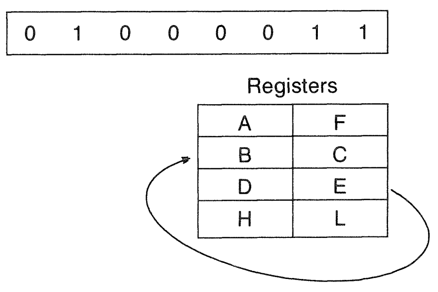
The following bit patterns determine the CPU registers. Refer to Fig. 8b.

ddd or sss	Register
111	A (Accumulator)
000	B
001	C
010	D
011	E
100	H
101	L

Machine cycle (M): 2, states (T): 7
Addressing: Register
Flags: Unchanged

"(r) ← (r')" means that the content of register r' is moved to register r.

If r and r' are B and E respectively, the bits become as follows:



Machine cycle: 1, State (T): 4

Transfer between a register and a memory

(2) Instruction: **LD A, (nn)**
Comment: Load accumulator direct
(A) ← (Byte 3) (Byte 2)
nn: 16-bit value in the range
0 — 65535

Machine word

Byte 1	0 0 1 1 1 0 1 0	OP code
Byte 2	Low-order address n	Address number
Byte 3	High-order address n	

The content of the memory locations, whose address is specified in the bytes 2 and 3, is moved to Accumulator. Here the Byte 2 carries the low-order address information and the Byte 3 carries the high-order address.

First, the operation code is applied to the Instruction Decoder and is decoded, the CPU understands that the next two bytes are address data then the content of the memory (01110110 for example) is loaded in the Accumulator.

M: 4, T: 13
Addressing: Direct
Flag: Unchanged

If the (nn) is 3241 (H) (#12865 (D)),

Instruction	Machine language	Hex format
LD A, (nn)	0 0 1 1 1 0 1 0	3A (H)
Byte 2	0 1 0 0 0 0 0 1	41 (H)
Byte 3	0 0 1 1 0 0 1 0	32 (H)

	Content							
Memory Address #3241 (H) (#12865 (D))	0	1	1	1	0	1	1	0
	⬇							
Accumulator	0	1	1	1	0	1	1	0

*3241 (Hex) = 12865 (Decimal)
This requires 4 machine cycles to be executed.

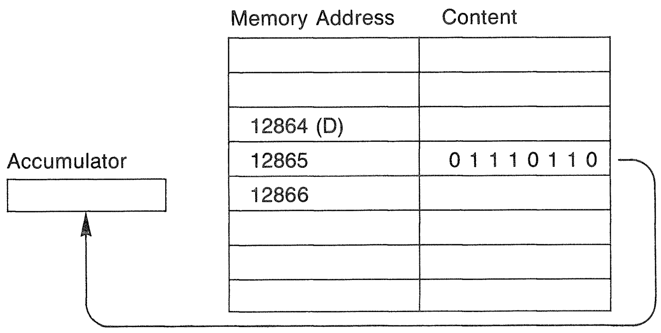


Fig. 45 Load the Content of the Memory Specified by Byte-2 & Byte-3 to Acc.

These instructions are in the direct addressing mode.

Direct addressing: Addressing the memory or register location by describing its address number after operation code in the program.

Ex: 3 Instruction: **LD r, n**
Comment: Move immediate register
(r) ← (byte 2)

Machine word:

Byte 1	0	0	d	d	d	1	1	0
Byte 2	data n							

In the immediate addressing mode*, the content of byte 2 is moved to register r.

If the register r is D and the data is 12,

	$\overbrace{\quad}^{D}$	
Byte 1	0 0 0 1 0 1 1 0	16 (H)
Byte 2	0 0 0 1 0 0 1 0	12 (H)

Now the data 12 is stored in the register D.

**Immediate addressing: In this mode Byte 2 does not have the location number of the memory or register but a piece of data.*

Therefore, the piece of data is move to the destination directly. This is used as a constant in a program because it is unmodifiable while executing the program.

M: 2, T: 7

(4) Instruction: **LD r, (HL)**
Comment: Move memory to register
(r) ← ((H) (L))

Move the content of the memory location, whose address is in registers H and L, to register r.

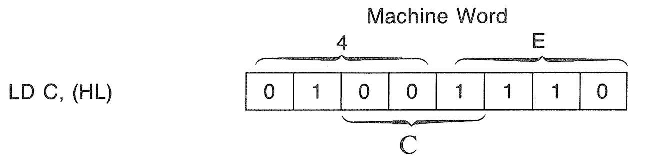
Machine word

0	1	d	d	d	1	1	0

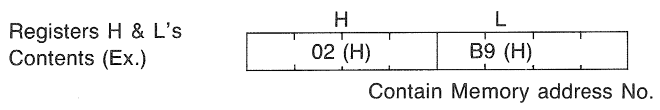
Instruction	Hex Format
LD A, (HL)	7E
LD B, (HL)	46
LD C, (HL)	4E
LD D, (HL)	56
LD E, (HL)	5E
LD H, (HL)	66
LD L, (HL)	6E

M: 2, T: 7
Addressing: Register indirect
Flag: Unchanged

If the r is C,



If the contents of registers H and L are 02 and B9 (H),



If the content of the memory #02B9 (H), is 0 0 1 1 0 1 1 1,

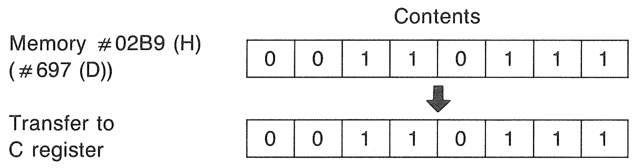


Fig. 46 Move the Content of the Memory Addressed by H & L Registers to Register C.

(5) Instruction: **LD r, (IX + d)**
Comment: Load the content of the memory #(IX + d) to the register r.

The value represented by "d" is added to the content of Index Register. The result becomes the real address to be called. The d may be −128 to +127.

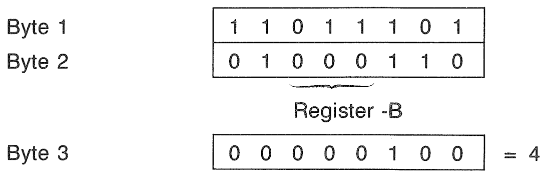
Machine word:

Byte 1	1 1 0 1 1 1 0 1
Byte 2	0 1 r r r 1 1 0
Byte 3	d

This facility makes the program code relocation possible without altering the code and is useful in handling elements in arrays and tables.

M: 5, T: 19
Flag: Unchanged

If the content of the IX register is 8002 (H), the value of "d" is 4, and "r" is B-register, the memory location of 8006 (8002+4) is addressed and the content of the memory #8006 is loaded in the B-register.



Content of IX	8002 (H)
Value of "d"	4 (H)
Addressed memory	#8006 (H)
Content of memory	0035 (H) for example
Load the content to	⬇
Register "r": B	0035 (H)

Indexed addressing: In this mode the address part is modified by an index register or similar device. This is a kind of indirect addressing. (5)

Indirect (relative) addressing: (3) ~ (5) the instruction does not state the address number of the memory to be accessed but indicates the register that stores the number.

5.2 (b) Arithmetic and logic instructions

Addition

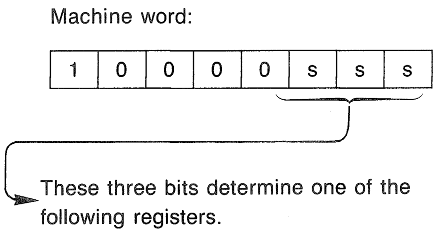
The most basic arithmetic function is the addition of two numbers.

Instruction: **ADD A, r**

Comment: Add register to A

(A) ← (A) + (r)

This instruction adds the content of the specified register to the content of the accumulator and stores the result in the accumulator.



Instruction	Register	Hex format
ADD A,	A	87
ADD A,	B	80
ADD A,	C	81
ADD A,	D	82
ADD A,	E	83
ADD A,	H	84
ADD A,	L	85

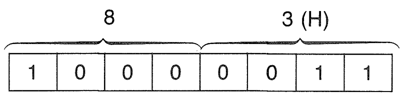
M: 1, T: 4

Addressing: Register

Flag: CY, Z, S and H are affected.
V = 1 if the result exceeds −128 or +127 (D)

If the result overflows, the carry flag becomes true. The flag acts as the ninth bit of the accumulator. The carry flag is also used when adding numbers longer than eight bits. In such a case, two or more registers are used to represent each number. The least significant bytes are added first. Then the most significant bytes are added. The carry from the least-significant bytes are also added. The carry bit works as a link between the two bytes.

Ex: ADD A, E



		Contents		
Register E		1 0 1 0	0 0 1 1	.... 163 (D)
Accumulator	+)	1 0 1 1	0 0 1 0	.... 178 (D)
″		1 0 1 0 1	0 1 0 1	.... 341 (D)
		Carry		

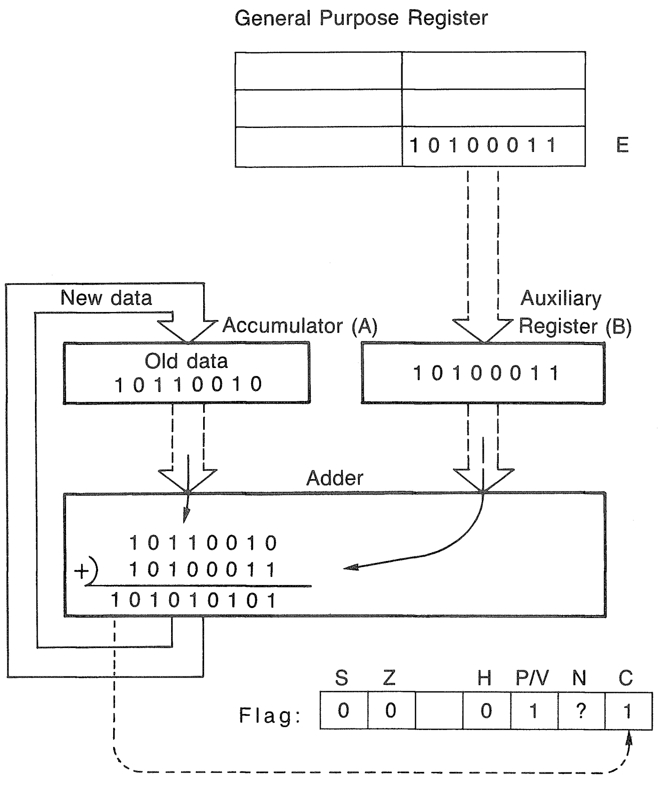


Fig. 47 Add the Content of Accumulator with the Content of Register-E

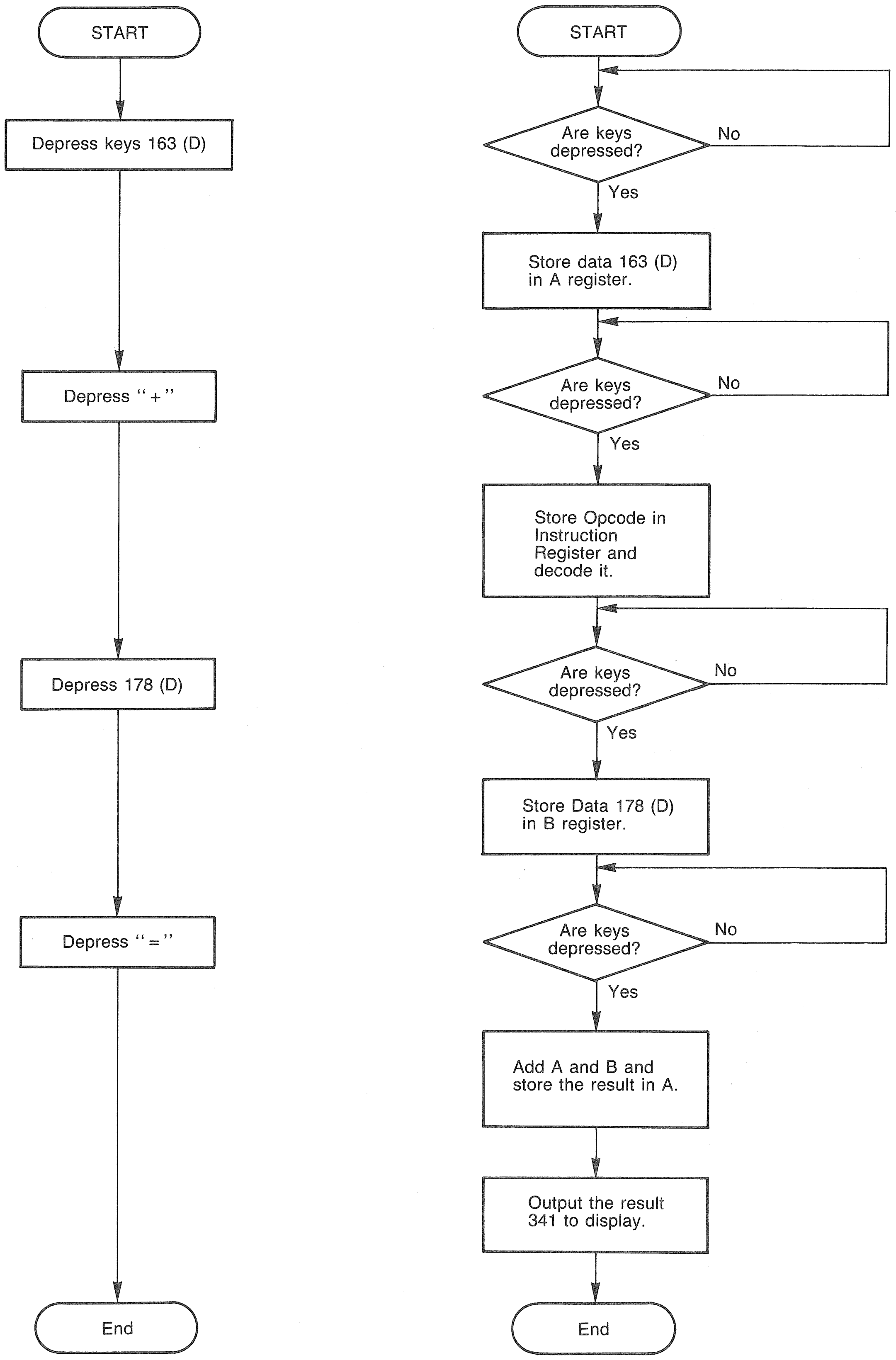

START
Depress keys 163 (D)
Depress "+"
Depress 178 (D)
Depress "="
End
START
Are keys depressed?
No
Yes
Store data 163 (D) in A register.
Are keys depressed?
No
Yes
Store Opcode in Instruction Register and decode it.
Are keys depressed?
No
Yes
Store Data 178 (D) in B register.
Are keys depressed?
No
Yes
Add A and B and store the result in A.
Output the result 341 to display.
End

Fig. 48 Flowchart for Executing 163 + 178 = 341 (D)

Instruction: **DAA**
Comment: Decimal adjust A

Machine word:

0	0	1	0	0	1	1	1

To meet the number system of the peripheral device, the 8-bit number in the accumulator is adjusted to form two 4-bit BCD digits as follows:

If the value of the least significant 4 bits (or the most significant 4 bits) of the accumulator is greater than 9 or if the H flag (half carry or auxiliary carry) is set, 6 is added to the 4 bits. Then 1 is carried up. In this case, all flags are affected.

M: 1, T: 4
Flag: S, Z, H, P and C are affected.↗

Ex: 1 25 + 39 = 64 (D)

```
                                 2       5
Accumulator                    0010    0101
                                 3       9
Temp register              +)  0011    1001
                           --------------------
                                 5       E
Result of binary add           0101    1110     Carry= 0,
                                 0       6         H= 1
DAA (+06)                  +)  0000    0110
                           --------------------
                                 6       4
Result                         0110    0100
```

Subtraction

Subtraction is carried out by adding the 2's complement of the subtrahend to the minuend. Refer to 1.4.

Instruction: **SUB A, (HL)**
Comment: Subtract memory from A
A ← (A) −((H) (L))

Machine word

1	0	0	1	0	1	1	0

M: 2, T: 7

This word means that subtract the content of the memory location, whose address is contained in the H and L registers, from the content of the accumulator, and place the result in the accumulator.

When the processor understands the meaning of the instruction, it addresses the memory of the number stored in the H and L registers, fetches the content of the memory to the temporary register, converts the content of the temporary register into 2's complement, adds it to the content of the accumulator, and finally stores the result in the accumulator. If the carry flag becomes "1", the result is considered to be a positive number, and if it is "0", it is considered to be negative and is converted to 2's complement again.

Ex: 1 6 − 2 =4

```
Temporary register             00000010      ...... 2

Inverted                       11111101
1 is added                 +)  00000001
                           ----------------
2's complement                 11111110
Accumulator                +)  00000110      ...... 6
                           ----------------
Result                     +  100000100      .....+4
                           ↑  ↑
                        Sign  Carry
```

The Carry flag "1" indicates that the result is positive.

Ex: 2 2 − 6 = 4

```
Temporary register             00000110      ...... 6

Inverted                       11111001
1 is added                 +)  00000001
                           ----------------
2's complement                 11111010
Accumulator                +)  00000010      ...... 2
                           ----------------
Sum                       ──→ 011111100
                   Carry       ¦
                       ┌-------┘
If Carry is 0,         ¦
the result is          ¦       00000011
Inverted and 1         ¦   +)  00000001
is added.              ¦   ----------------
Result                 └--→ −  00000100      ...... 4
```

The Carry flag "0" indicates that the result is negative.

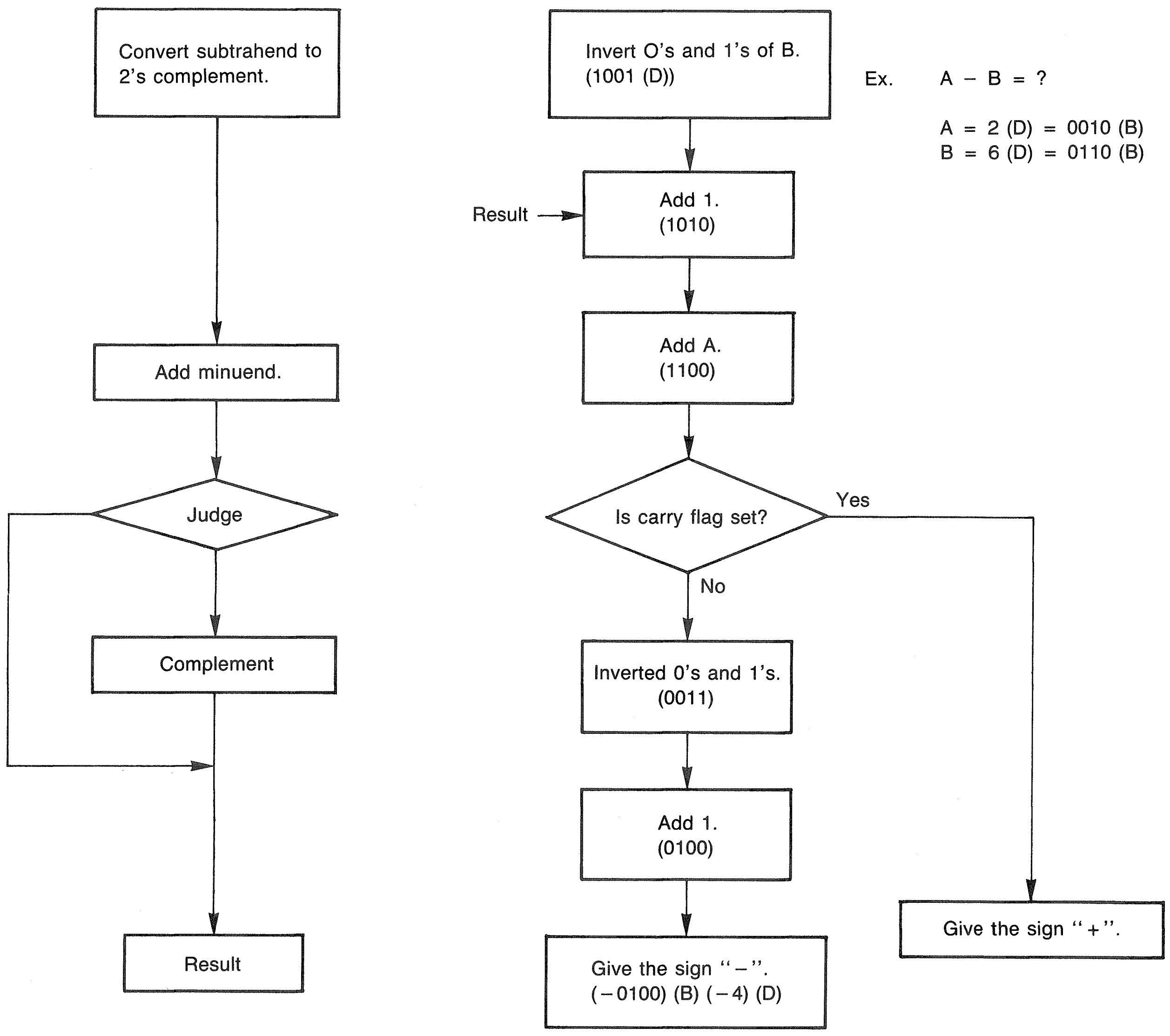


Fig. 49 Flow Chart for Executing "2 −6"

Logical instructions

Usually, these instructions affect the Zero, Sign, Parity, Auxiliary Carry and Carry flags.

Instruction: **AND r**
Comment: And register with A
(A) ← (A) ∧ (r) ∧: AND

Machine word:

1	0	1	0	0	s	s	s

M: 1, T:4
Addressing: Register
Flags:

S	Z		H		P	N	C
*	*		1		P	O	O

Affected

If the r is the register D, sss becomes 0 1 0.

1	0	1	0	0	0	1	0

This instruction causes the content of a register to be ANDed with that of the accumulator. The result is placed in the accumulator and the content of the other register remains unchanged. The ANDing is performed between the equally significant bits. The C-flag is cleared and H flag is set.

```
D-register      0 0 1 1 1 1 1 0  Remains unchanged
Accumulator   ∧ 1 0 1 1 0 0 1 1
"               0 0 1 1 0 0 1 0  Result
```

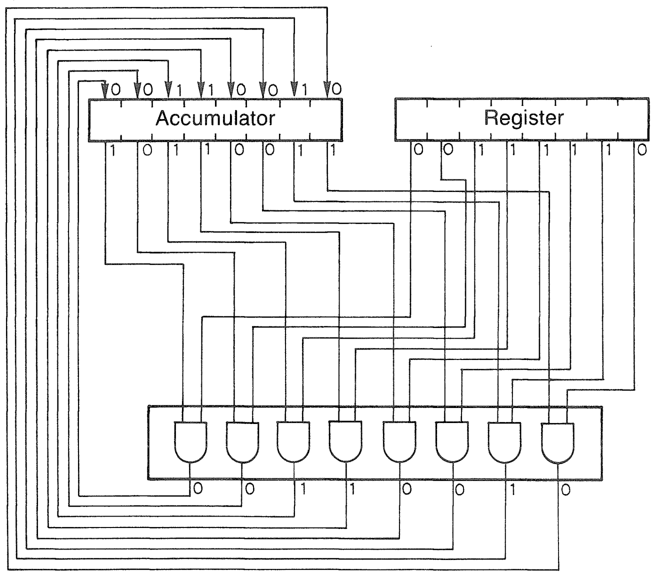


Fig. 50 ANDing the Contents of Accumulator and a Register

Instruction: **OR A, n**
Comment: Or immediate with A
(A) ← (A) ∨ (byte 2) ∨: OR

Inclusive-OR the content of byte 2 with that of the accumulator and place the result in the accumulator. The C and H flags are cleared.

Machine words:

Byte 1	1	0	1	1	0	1	1	0
Byte 2	data n							

M:2, T: 7

Instruction: **CP r**
Comment: Compare register with A
(A) — (r)

Machine word:

1	0	1	1	1	r

The content of register r is subtracted from that of the accumulator. The flags are as follows:

If (A) = (r),	Z	← 1
If (A) < (r),	CY	← 1

M: 1, T:4
Addressing: Register
Flags:

S	Z		H		P/V	N	C
*	*		*		V	1	*

If r is C,

1	0	1	1	1	0	0	1

5.2 (c) Branch instructions

Usually the program stored in the memory is executed sequentially. This group of instructions alters the normal program flow. Condition flags are not affected by any of the instructions. Unconditional branches simply alter the content of the Program Counter and thus the sequence of the program flow. Conditional transfers branch only when one of the following conditions is satisfied:

Condition		ccc
NZ — Not zero	(Z = 0)	000
Z — Zero	(Z = 1)	001
NC — No carry	(CY = 0)	010
C — Carry	(CY = 1)	011
PO — Parity odd	(P = 0)	100
PE — Parity even	(P = 1)	101
P — Plus	(S = 0)	110
M — Minus	(S = 1)	111

5.3 (c) Branch Instruction

Instruction: **CALL nn**
Comment: Call unconditional
((SP) − 1) ← (PCH)
((SP) − 2) ← (PCL)
(SP) ← (SP) − 2
(PS) ← (byte 3) (byte 2)

Machine words:

Byte 1	1 1 0 0 1 1 0 1
Byte 2	Low-order address n
Byte 3	High-order address n

The content of Stack Pointer (SP) is decremented by one. The high-order eight bits of the next instruction address are moved to the memory location whose address is specified by SP. The content of SP is decrement by one again. The low-order eight bits of the address are moved to the memory location whose address is in SP. Control is transferred to the instruction whose address is specified in bytes 3 and 2 of the current instruction.

M= 5, T= 17
Addressing: Immediate/reg. indirect
Flags: Unchanged
Ex: CALL 8572 (H)

		(P Counter)
Byte 1	1 1 0 0 1 1 0 1	-----8001
Byte 2	0 1 1 1 0 0 1 0	--72--8002 (H)
Byte 3	1 0 0 0 0 1 0 1	--85--8003
		-----8004

		(S Pointer)
		9499 (H)
Address to be returned	80	----- 9498
	04	----- 9497
		----- 9496

Instruction: **RET**
Comment: Return
(PCL) ← ((SP))
(PCH) ← ((SP) + 1):
(SP) ← (SP) + 2

Machine word: 1 1 0 0 1 0 0 1

The content of the memory location whose address is specified by SP is moved to the low-order eight bits of PC. The content of SP is incremented. The content of the memory next to the above is moved to the high-order bits of PC. The content of SP is incremented by one again. The operation is reverse to the CALL instruction.

M= 3, T= 10
Addressing: Register indirect
Flags: Unchanged

Instruction: **JP nn**
Comment: Jump unconditional
(PC) ← (byte 3) (byte-2)

Machine words:

Byte 1	1 1 0 0 0 0 1 1
Byte 2	Low-order address n
Byte 3	High-order address n

Control is transferred to the instruction whose address is specified by bytes 3 and 2 of the current instruction. No data are pushed on the Stack.

M= 3, T= 10
Addressing: Immediate
Flag: Unchanged

5.4 (d) (e) Stack, I/O and Machine Control Instructions

Unless otherwise specified, condition flags are not affected by any instructions in this group.

Instruction: **PUSH qq**
Comment: Push register pair on stack

Machine words: 1 1 q q 0 1 0 1

((SP) − 1) ← (rh) 1
((SP) − 2) ← (rl) 2
(SP) ← (SP) − 2

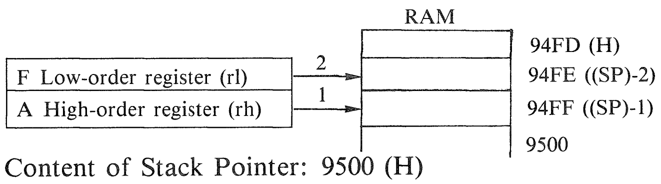
If you want to change your job, you have to put the materials and data aside on a shelf. In the computer you have to do the same job. This instruction means that: Move the content of the high-order register of a register pair (rp) to the memory location whose address is one less than the content of Stack Pointer (SP). Move the content of the rp's low-order register to the memory location whose address is two less than the content of SP. Decriment the content of SP by 2.

Instruction	Hex	qq	bits
PUSH BC	C5	00	Push rp (B&C) on stack
PUSH DE	D5	01	Push rp (D&E) on stack
PUSH HL	E5	10	Push rp (H&L) on stack

M= 3, T= 11
Addressing: Indirect
Flags: Unchaged

If you want to store the contents of the accumulator and flag register in the stack area in a RAM,

Ex: Instruction: **PUSH AF**
Machine word: | 1 1 1 1 0 1 0 1 |



Content of Stack Pointer: 9500 (H)
↓
94FE (SP)-2

Instruction: **POP qq**
Comment: Pop to register pair
Machine word: | 1 | 1 | q | q | 0 | 0 | 0 | 1 |

(rl) ← ((SP))
(rh) ← ((SP) + 1)
(SP) ← (SP) + 2

This requires the processor to bring the stacked information back to the register pair in the reverse way to the PUSH operation. The content of SP is incremented by 2.

M= 3, T= 10
AddressingIndirect
Flags: May change when POP AF↗

Instruction: **OUT (n), A**
Comment: Output to port n
$A_7 \sim A_0 \leftarrow n$
$A_{15} \sim A_8 \leftarrow A$
$(n) \leftarrow A$

n: Port number
(n): Data to be moved to port n

Place the content of register A on the data bus for transmission to a specified port.

Machine words	1 1 0 1 0 0 1 1
Byte 2	n (Port No.)

M= 3, T= 11
Addressing: Direct
Flags: Unchanged

Instruction: **NOP**
Comment: No op
Machine word: | 0 0 0 0 0 0 0 0 |

No operation is performed with this instruction. The CPU resumes its operation after 1μs (0.25μs × 4T).

M= 1, T= 4
This instruction is often used to make a blank space in the program for future use or to delete unnecessary instructions.

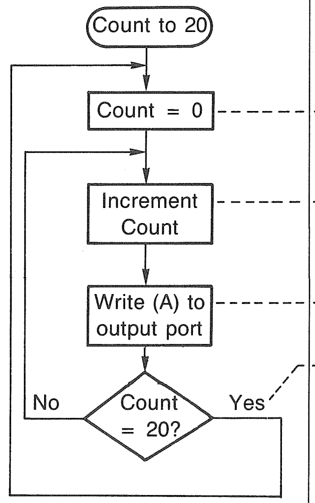
7. Programming

A computer requires a program or a string of instructions to perform a job. The instructions should written in the right order because a microcomputer can not arrange the order.

Ex. Count from 0 to 20 (D).

Fig. 51 shows the flow chart for counting and displaying from 0 to 20 repeatedly and its program written by assembly language and machine language. Each instruction consists of an operation code (opcode) and memory address or data. The code comes first and the data or address next. Each opcode determines whether it should be followed by data or an address. The label provides the same function as the line number. Jumping to the labeled line is possible only by referring to the label name.

First, "LD A, 0" clears Accumulator (A) by loading data "0" to it. Next, "INC A" increments the value in the A. This instruction requires no data. Next, at the line #3, the A's content is put out to Port #7. This is a 2-byte instruction. The Byte-2 defines the output-port number; 7. "CP A, 20" compares the value in A with 20. If the value is equal to 20, a zero flag is set. "JP Z" tests this flag. If the flag is set, it causes a jump to the line labeled START. The 2-byte data following "JP Z" opcode carry the address for jumping. Note that the low-order address is stored first and the high-order address next. JP is the unconditional jump instruction while "JP Z" is conditional. The microprocessor reads the note, LOOP, and understands where to jump. JP LOOP is similar to JP Z START. When the microprocessor fetches the JMP LOOP instruction, it returns to the address labled as LOOP.

Flow Chart	Line No.	Z80A Assembly Language		Z80A Machine Language		
				Memory		
		Label Instruction	Comment	Address	Contents	
				Hex	Hex	Binary
	1	START: LD A, 0	; Set A register to 0	8000	3E	00111110
			(data 0)	8001	00	00000000
	2	LOOP: INC A	; Increment A register (1-byte instruction)	8002	3C	00111100
	3	OUT (7_H), A	; Output A to Port #7	8003	D3	11010011
			(Port No.)	8004	07	00000111
	4	CP A, 20	; Compare A register with 20. (A − 20 = ?)	8005	FE	11111110
				8006	14	00010100
	5	JP Z START	; Go to beginning if A = 20	8007	CA	11001010
			(Address 8000)	8008	00	00000000
				8009	80	10000000
	6	JP LOOP	; Repeat (Jump to LOOP)	800A	C3	11000011
			(Address 8002)	800B	02	00000010
				800C	80	10000000

(A): Content of accumulator

Fig. 51 Flow Chart and Assembly/Machine Languages for Counting and Displaying 0 to 20.

7.1 Changing program flow

The PC first indicates the top address (8001) of the memory of a program. Usually CPU fetches instructions of the program and execute them one by one incrementing the content of PC. So, the instructions should be written in the right order.

Before completing the main program, other program such as interrupt service and subroutine can be performed. The CPU checks if a Hold, Interrupt or Halt signal appears every time when a machine cycle is completed.

7.2 Interrupt

Accidents such as fire may occur while proceeding the main program. In such a case a fire extinguishing job may have to be done right away. Interruption is possible automatically with a logic circuit or manually. When an Interrupt request is acknowledged, the main program is put aside till the interrupt subroutine is completed. Z80A can perform up to five interrupt subroutines because it has five interrupt inputs. The subroutine programs can be stored in a RAM. Each interrupt is assigned a priority. The program of the highest priority is acknowledged first when more than two interrupts are requested. Without the ↗ interrupt programs the interrupt signals will be ignored. There are two kinds in interrupt requests: $\overline{\text{NMI}}$, (non maskable interrupt) and $\overline{\text{INT}}$ (maskable interrupt).

The highest priority is given to $\overline{\text{NMI}}$ signal because this is used for taking an emergency measure against a power failure, fire, etc. When the power goes off and $\overline{\text{NMI}}$ signal is set, the processor interrupts the current program and accepts $\overline{\text{NMI}}$ service routine. The $\overline{\text{INT}}$ request is accepted provided that the interrupt enable flip-flop (IFF) is set.

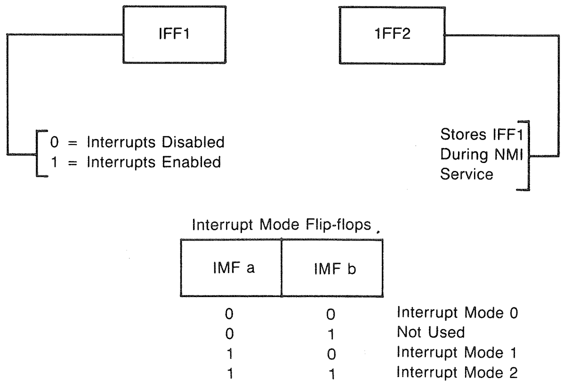


Fig. 52 Interrupt Flip-flops Status

Stack operation

Fig. 53 shows how a program flow is changed when an $\overline{\text{INT}}$errupt is requested. Prior to start an $\overline{\text{INT}}$ service, the interim result and other information of the sustained main job such as the contents of PC, Accumulator and registers concerned should be stored somewhere till the $\overline{\text{INT}}$ service is completed. Some CPUs have internal stack registers for storing them. Z80A uses external memories for the stack and a stack pointer for indicating the address number of the stack memory. Let's see the process in the example of Fig. 53.

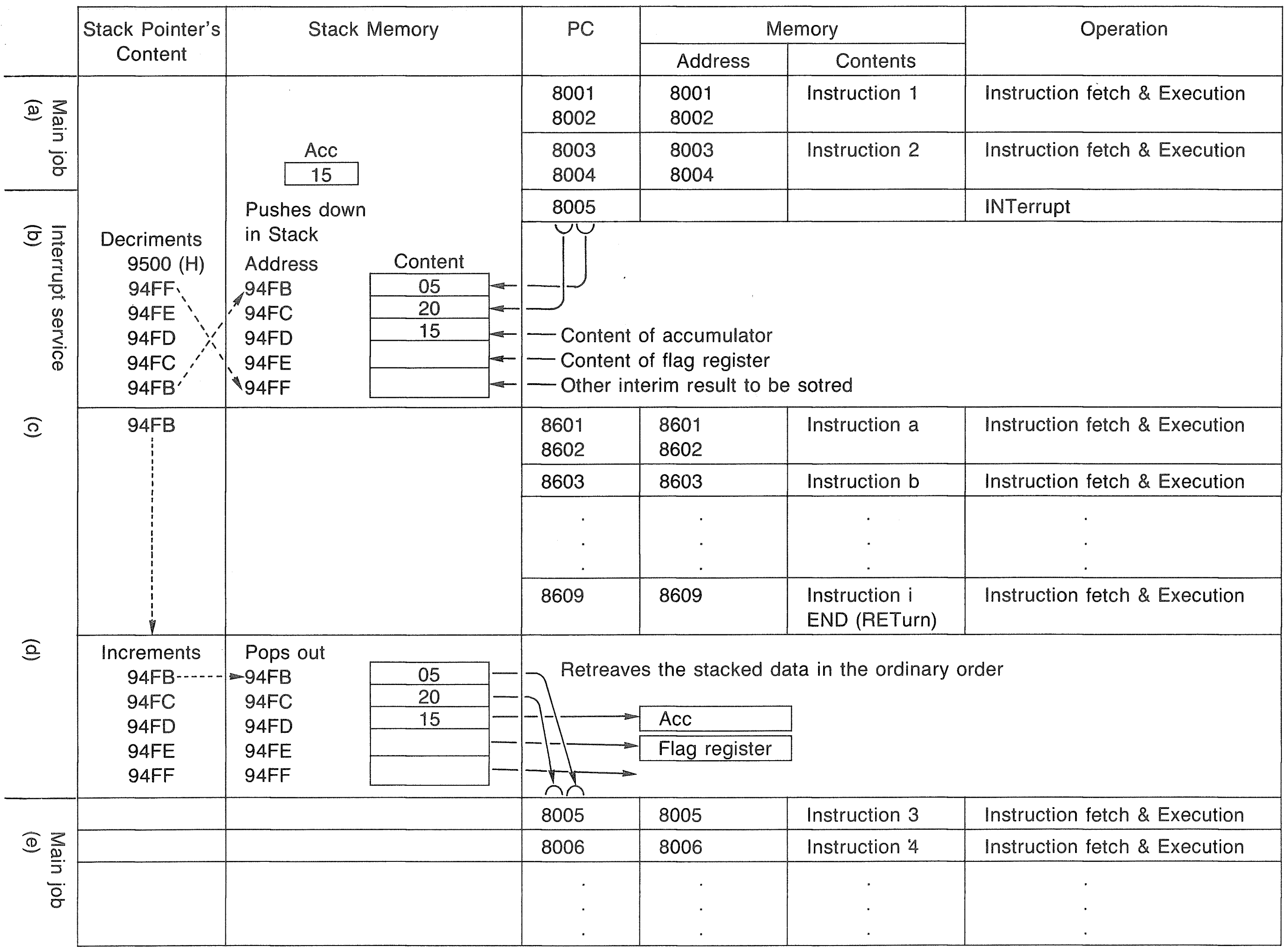
a) If the $\overline{\text{INT}}$ signal becomes true (L) when the instruction 2 is fetched, the CPU acknowledges it at the end of the current machine cycle, and then the PC counts up to 8005.

b) The main program is sustained at 8004. The contents of PC (8005), accumulator (15), flag register and other necessary data are pushed into the stack memory space in turn, and 8601 is loaded in the PC.

c) $\overline{\text{INT}}$ service is started from the instruction at the address #8601. The PC shifts from 8601 to 8609.

d) When the $\overline{\text{INT}}$ service is finished, the data stored in the stack are retrieved to the PC, ACC, flag register, etc. in the reverse order to that of pushing.

e) The PC is loaded with 8005, the next address of the memory of the sustained main program, and the CPU resumes the main program.



Stack Pointer's Content
Stack Memory
PC
Memory
Address
Contents
Operation
(a) Main job
8001 8002 | 8001 8002 | Instruction 1 | Instruction fetch & Execution
Acc 15
8003 8004 | 8003 8004 | Instruction 2 | Instruction fetch & Execution
8005 | INTerrupt
(b) Interrupt service
Decriments 9500 (H) 94FF 94FE 94FD 94FC 94FB
Pushes down in Stack
Address 94FB 94FC 94FD 94FE 94FF
Content 05 20 15
Content of accumulator
Content of flag register
Other interim result to be sotred
(c) 94FB
8601 8602 | 8601 8602 | Instruction a | Instruction fetch & Execution
8603 | 8603 | Instruction b | Instruction fetch & Execution
8609 | 8609 | Instruction i END (RETurn) | Instruction fetch & Execution
(d) Increments 94FB 94FC 94FD 94FE 94FF
Pops out 94FB 94FC 94FD 94FE 94FF
05 20 15
Retreaves the stacked data in the ordinary order
Acc
Flag register
(e) Main job
8005 | 8005 | Instruction 3 | Instruction fetch & Execution
8006 | 8006 | Instruction 4 | Instruction fetch & Execution

Fig. 53 Program Flow of Interrupt Service

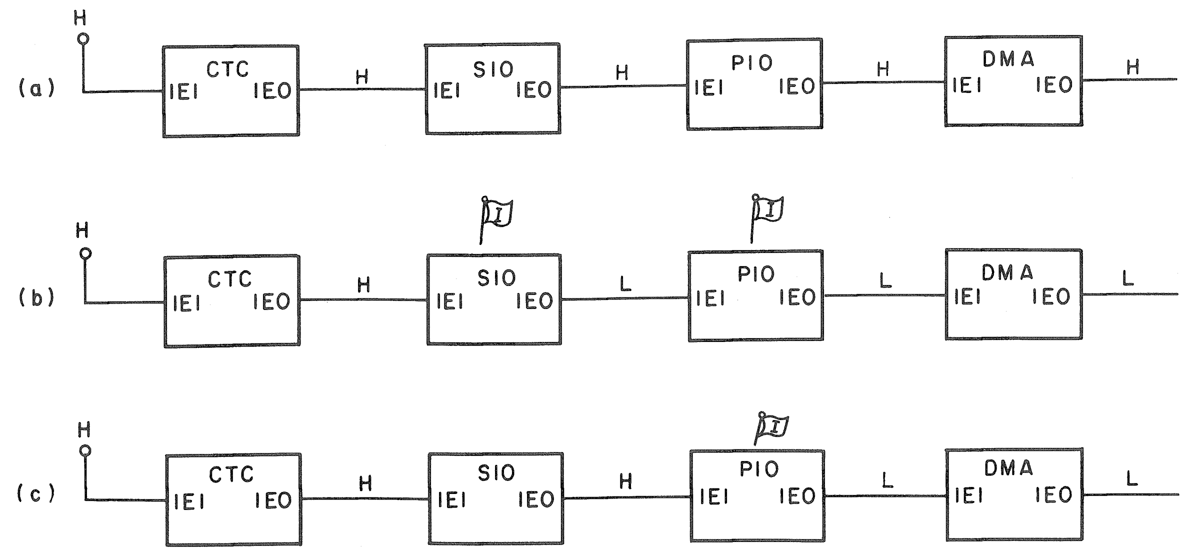
Priority of interrupts

Z80A's peripheral LSI has a logic circuit which makes Mode-2 interrupt (Daisy chain) service possible. So there is no special LSI for controlling interrupts. The priority of interrupts is determined by the Daisy chain system.

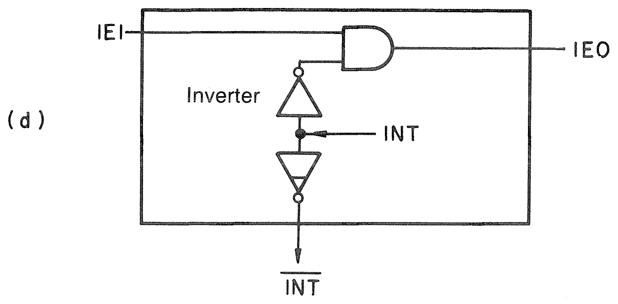
Daisy chain system

In Fig. 54 the interface chips are connected serially. The IEO (Interrupt Enable Output) of the left side chip is connected with the IEI (Interrupt Enable Input) of the right one. The voltage on the interconnecting lines is kept high (H) when no interrupt is requested as shown in Fig. (a). The interrupt request of each chip is acknowledged only when its IEI is H, and it shifts its IEO to L when requesting an interrupt. It maintains its IEO L till its interrupt service is completed. Each chip maintains its IEO L when IEI is L.

The IEI of the leftmost chip is always kept H. In this way, the order of priority is given serially from the left to the right. In Fig. (b), if the SIO and PIO request an interrupt at the same time, the SIO shifts its IEO to L and prohibits PIO's request. The SIO's request is acknowledged by the CPU and its interrupt service is executed. The SIO's request is cleared when the RETI instruction at the end of the interrupt program is executed. Then the SIO's IEO shifts to H to allow the PIO, the chip of the next priority, to request its interrupt → Fig. (c). Fig. (d) shows that the chip puts out low IEO signal when IEI is L , and when an interrupt is requested, the high INT signal is inverted and lets the AND circuit put out low IEO.



(a) No chip is requesting interrupt.
(b) Two chips are requesting it at the same time.
(c) SIO finished its job and the interrupt request of the PIO is acknowledged.



IEI: Interrupt Enable Input
IEO: Interrupt Enable Output

CTC: Counter Timer Circuit
SIO: Serial Input-Output interface
PIO: Parallel Input-Output interface

Fig. 54 Daisy Chain System

7.3 CALL and JUMP

Programs for repetitive tasks can be put together by making a Subroutine Program. A subroutine is a portion of a program that may be branched to or called from other parts of the program. There are some differences between Call and Jump. The program in Fig. 55 (a), shows a Subroutine started by a CALL instruction and terminated by RETurn. When RET is executed, the PC is loaded with the next address number to that of CALL X and the sequence of the main program is resumed. The same Subroutine can be called for repeatedly in a main program.

The right program of the Fig. (b) is called by Jump (JMP) instruction and does the same job as that of Fig. (a). JMP X causes the CPU to jump to the subroutine and JMP Y to address #8003. At the next JMP X it jumps to #8500 and does the same job. When it finishes the routine, JUMP Y causes the flow to jump to #8003, not to #8006 because the JMP address is inflexible. JUMP instruction, therefore, can not be used for a subroutine which usually requires the control to be returned to more than one place in the master routine.

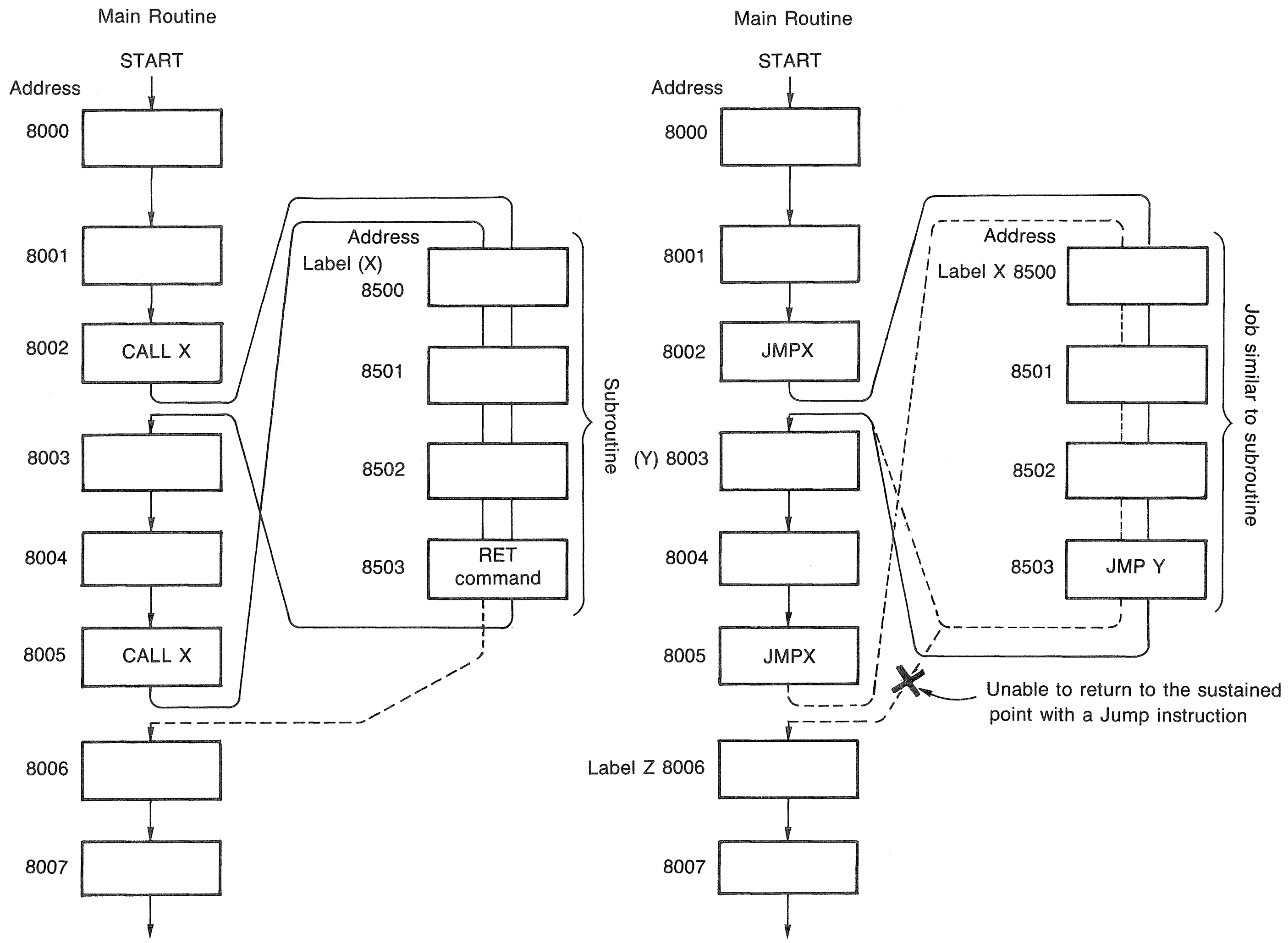


Fig. 55 CALL and JUMP Instructions

When writing a program for PX-7, use the memories in the User Area from 8000 (H) to F380 (H). When programming a machine language subroutine, first secure an area for subroutine by using the CLEAR statement. Then define the starting address with the DEFUSR statement.

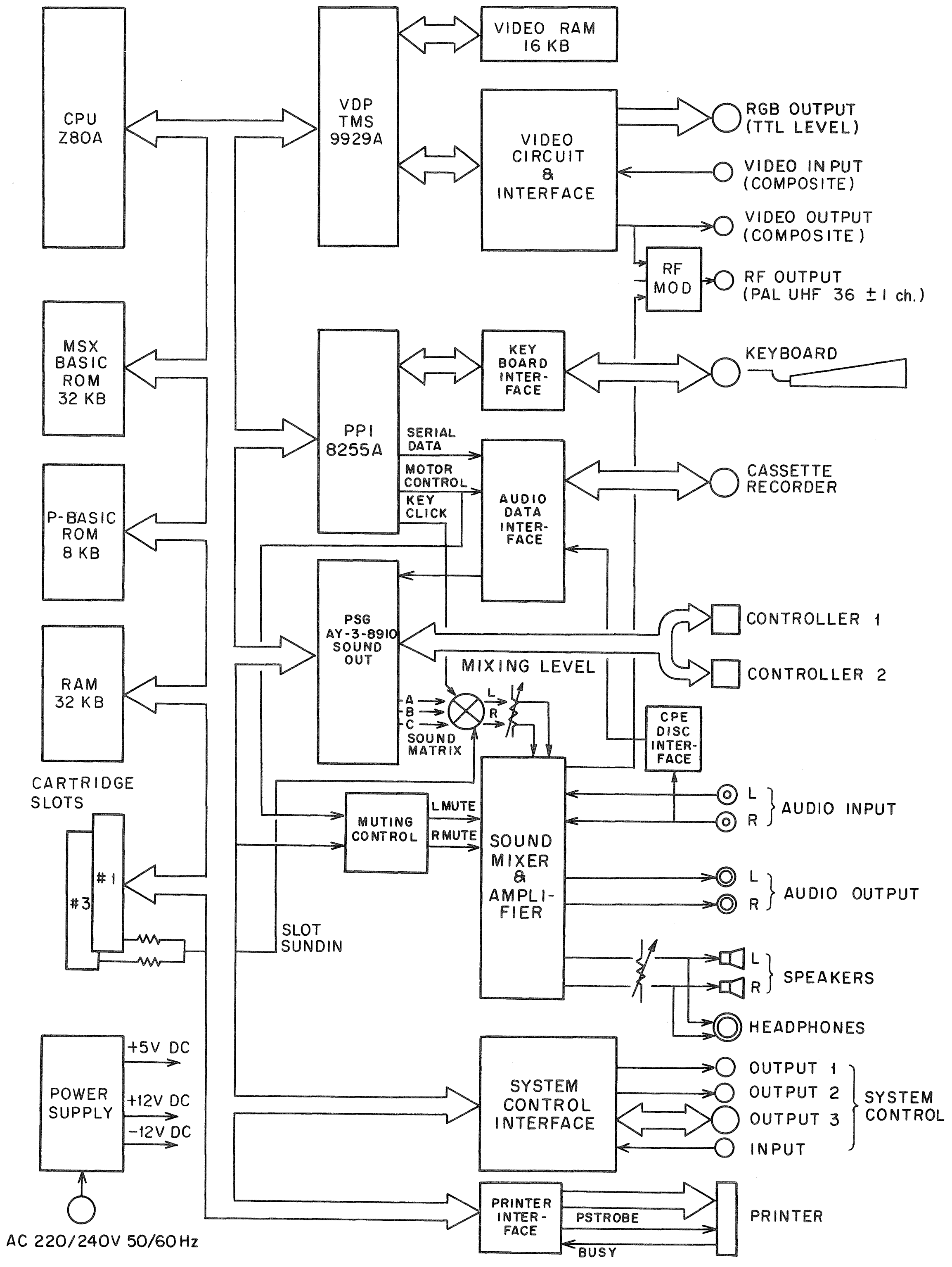


Fig. 56 Block Diagram of PX-7

For further details, refer to the operating manual of Zilog Z80A CPU and the operating instructions and service manual of PX-7.

By K. Satoh and S. Hirose

New Products

PAL-VHS Home VCR (VH-600) Continued

In the previous edition Vol. No. 8, we discussed NTSC VCRs. Here we will discuss PAL VHS VCRs basing on our model VH-600. The basic mechanism of VH-600/VHS of PAL version is the same as that of NTSC version. One turn of the drum of home VCR lets the CH-1 and CH-2 heads trace one frame. The revolution of PAL version drum, however, is 1,500rpm (25 frames × 60 sec.), the same speed as that of the PAL CAV LV disc. The difference of the writing speed between the PAL and NTSC versions is the difference of the rotational speed of the drum because the diameter of PAL drum is the same as that of NTSC.

Table 1 Main Specifications of PAL/NTSC TV Signals

	PAL	NTSC
Frames/sec.	25	30
Fields/sec.	50	60
Trace lines/frame	625	525
Scanning frequency (fH) [kHz]	15.625	15.734
Burst frequency [MHz]	4.4336	3.579545

Table 2 VHS PAL/NTSC Specifications for Standard Play (SP) Mode

		PAL	NTSC
Tape speed (mm/sec.)		23.39	33.35
Writing speed (mm/sec.)		4.87	5.8
Video track width (mm)		10.6	
Video track pitch (μm)		48.6	58
Video rec system		2-head azimuth	
Video head azimuth		±6°	
FM carrier (MHz)		3.8 — 4.8	3.4 — 4.4
Converted frequency (kHz)		(40+1/8)fH=627k	40fH = 629k
Phase shift (CH1)		0°	+90°
(CH2)		−90°	−90°
Drum rotation (rpm)		1,500	1,800
Audio FM carrier (MHz)	R	1.8	1.7
	L	1.4	1.3

1. C-signal Crosstalk Suppression

The process of cancelling crosstalk is a little different from that of NTSC because the polarity of the R-Y signal of PAL is inverted line by line and has no 1H-line or adjacent-line correlation kept by the NTSC system but has 2H-line correlation. Fig. 2 simplifies the principle of PAL VHS C-signal crosstalk cancelling system. It shifts the signal for Track-B by 90° every 1H when recording and returns it and adds the 2H-delayed signal with the non-delayed in playback.

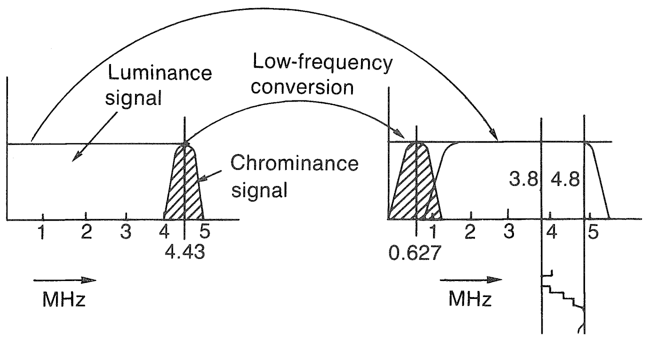
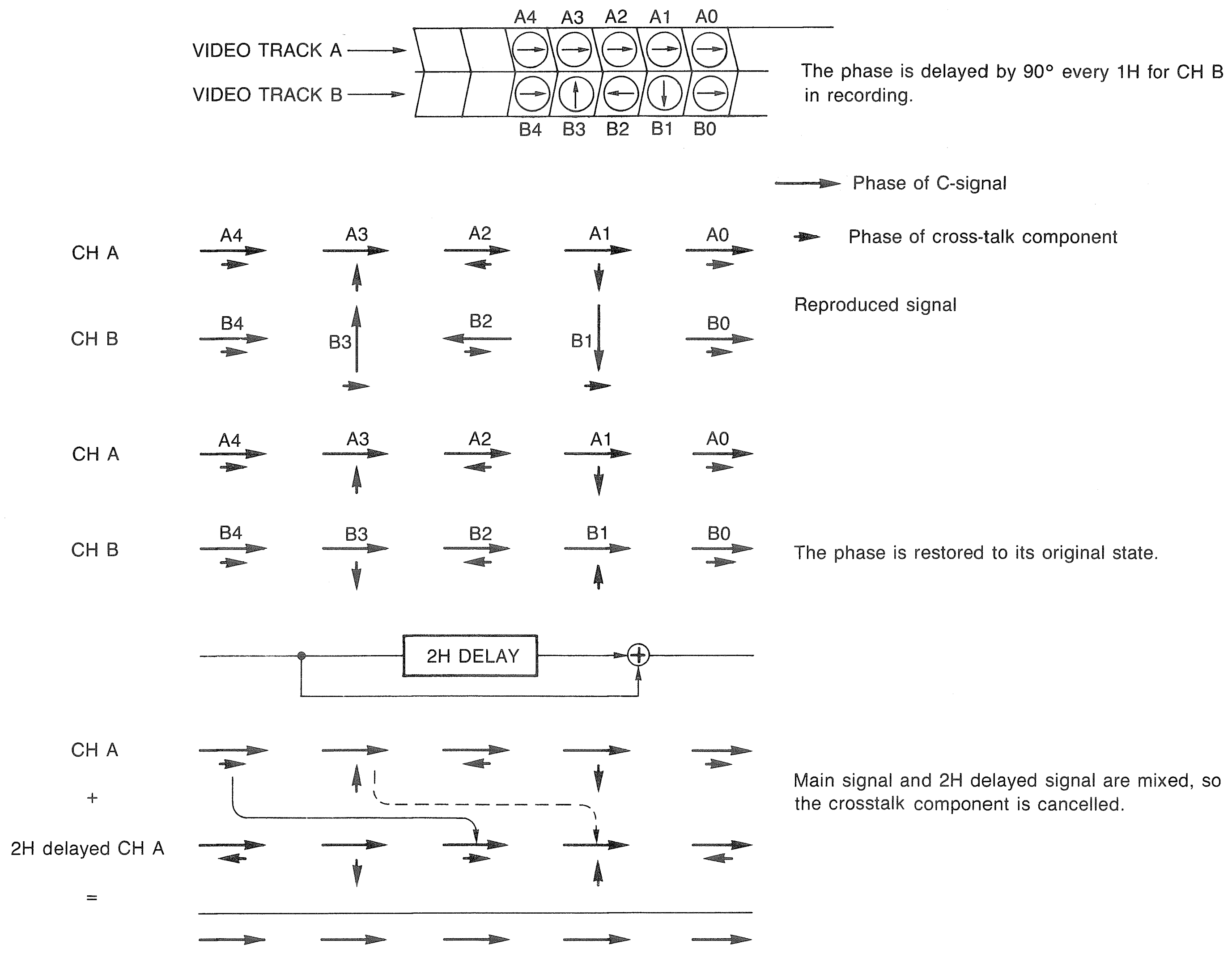


Fig. 1 Low-frequency-conversion of Chrominance Signal



A4 A3 A2 A1 A0
VIDEO TRACK A
VIDEO TRACK B
B4 B3 B2 B1 B0
The phase is delayed by 90° every 1H for CH B in recording.
Phase of C-signal
Phase of cross-talk component
CH A
CH B
Reproduced signal
CH A
CH B
The phase is restored to its original state.
2H DELAY
CH A
+
2H delayed CH A
=
Main signal and 2H delayed signal are mixed, so the crosstalk component is cancelled.

Fig. 2 C-signal Cross-talk Suppression

2. Video Circuit of VH-600/PAL

2.1 Block Diagram of Recording Circuit

The upper blocks from Automatic Gain Control (AGC) to 1.5MHz High-pass Filter (HPF) process Luminance (Y) signal and the lower blocks process C-signal.

Y-signal: Video signal level is made constant by AGC. Y-signal is taken out of the video signal by LPF. The Y-signal is equalized, clamped and then pre-emphasized. There are two kinds in pre-emphasizers, Dynamic Pre-emphasizer which increases its amplification in high frequency range as the input level decreases, and Main Pre-emphasizer which emphasizes the signal in high frequency range irrespective of the input level. Fig. 4 and 5 show the characteristics of the pre-emphasizers. Unwanted overshoot generated in pre-emphasizing process is clipped. The signal is then frequency-modulated (FM) in the range from 3.8MHz (sync tip) to 4.8MHz (white peak). Low frequency components are cut off by 1.5MHz HPF to make a room for 0.627MHz-converted C-signal.

C-signal: The 4.43MHz ± 500kHz C-signal filtered by BPF is made constant in level by Automatic Chrominance Control (ACC) and is applied to MAIN CONV and APC DET. APC DET extracts the 4.43MHz color burst from C-signal and compares the phase of the burst and 4.43MHz VCO signal. The error signal controls the VCO and locks its frequency to that of the burst signal coming from Video input.

The COMP SYNC separated from Y-signal is applied to (fH)/2 Killer where equalizing pulses are removed. AFC DET compares the phase of H-sync with that of the fH made by dividing 321fH VCO signal by 321. The error signal controls 321fH VCO (=5.06MHz). Then the 321fH-VCO is locked to the frequency of the H-sync taken from VIDEO IN. The 321fH-VCO signal is divided by 8 and becomes 627kHz (40 + 1/8)fH. The 4-phase-shifter delays the signal for recording on Tracks-B by 90°. SUB CONV mixes 4.43MHz VCO signal and the 627kHz signal. The added signal comes out of 5.06M BPF and is applied to MAIN CONV.

MAIN CONV mixes 4.43MH ± 500kHz C-signal and 5.06MHz (4.43M + 627kHz) signal. 1.5MHz LPF passes only the difference signal between the two. Now 4.43MHz ± 500kHz C-signal has been converted to low frequency signal. The low-frequency-converted C-signal is mixed with the Y-signal and is applied to REC Amp.

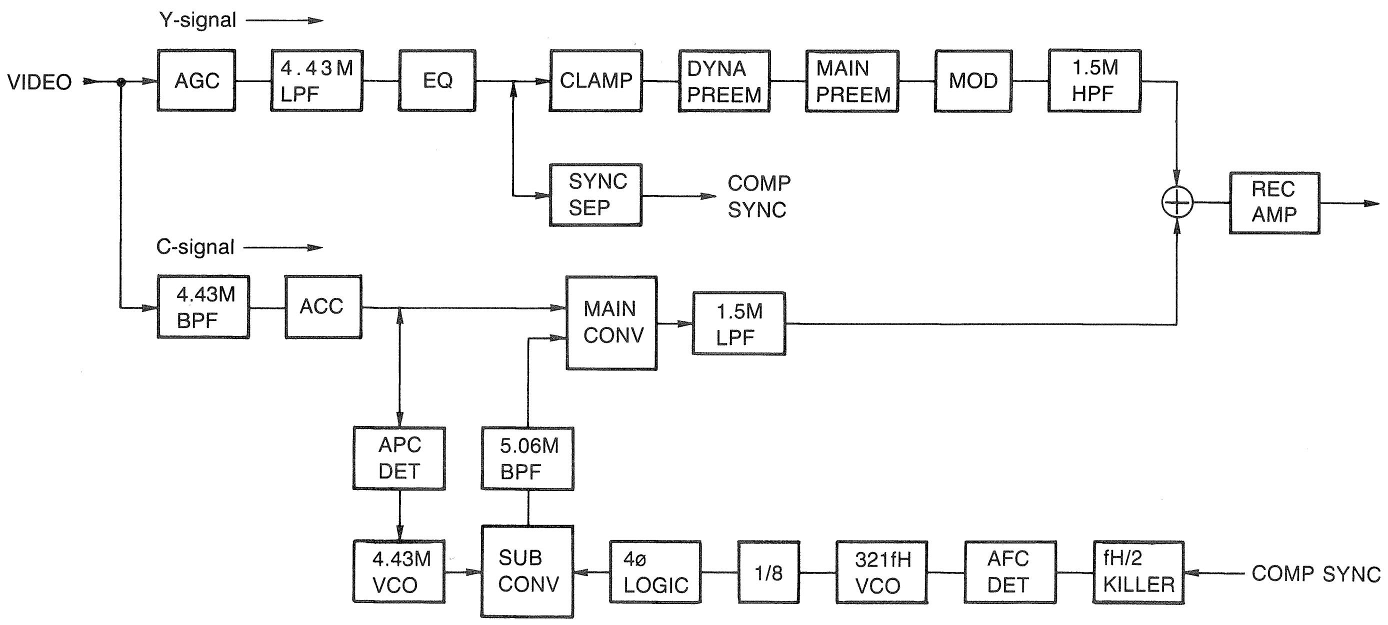


Fig. 3 Block Diagram of the Recording Circuit of VH-600/PAL

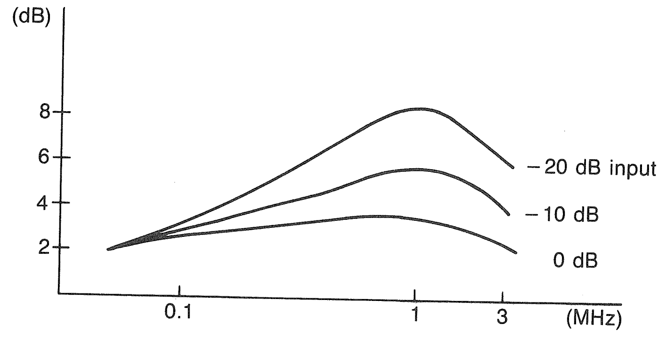


Fig. 4 Characteristics of Dynamic Pre-emphasis

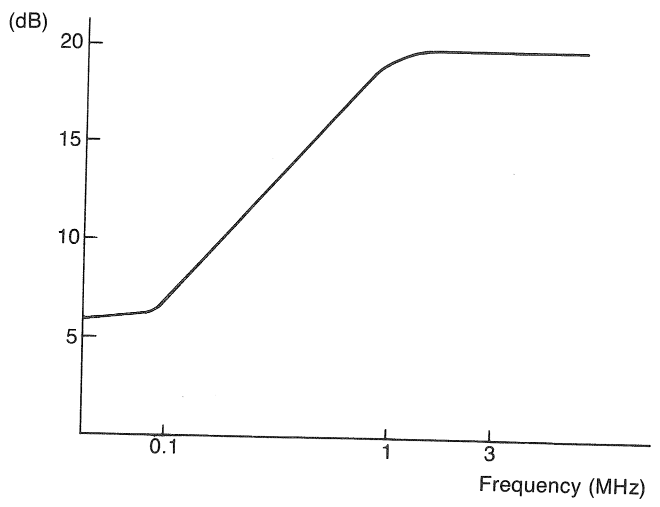


Fig. 5 Characteristics of Main Pre-emphasis

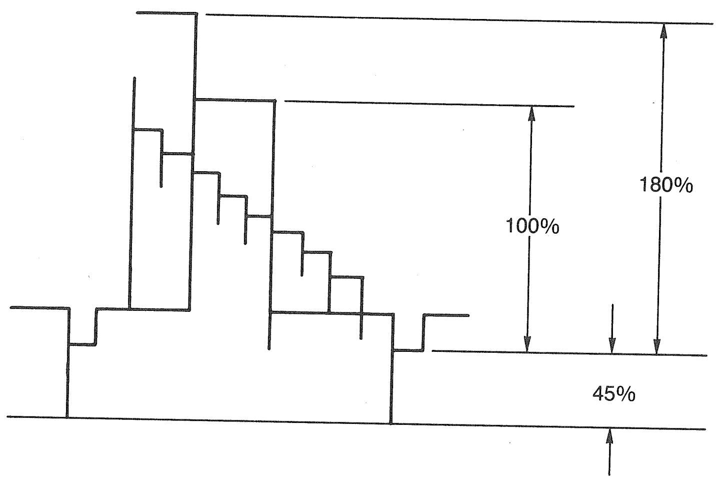


Fig. 6 Characteristics of Clip Circuit

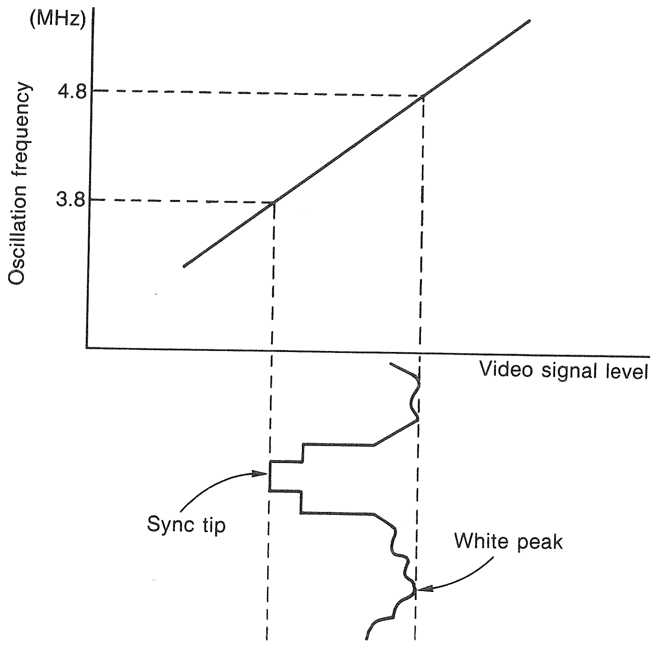


Fig. 7 Y-signal Frequency Modulation

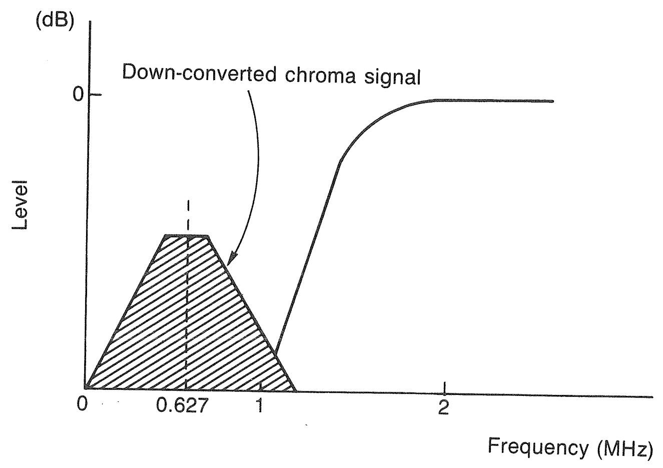


Fig. 8 Characteristics of 1.5M HPF

2.2 Y-signal Processing in PLAY Mode

The signals alternately picked up by the channel-A (CH-A) and channel-B (CH-B) heads are switched by a 25Hz square wave signal and are edited to become a continuous signal. Refer to the Fig. 10, p34, TUNING FORK 8. The reproduced signal contains the FMed Y-signal and low-frequency-converted C-signal. The Y-signal is taken out by 5MHz-Peak Equalizer, AGCed, dropout is compensated, limited and then demodulated. LPF removes FM carrier from the demodulated signal, and De-emphasizer decreases the emphasized signal level down to restore the original flat frequency characteristic. The signal is noise-cancelled, mixed with the reproduced C-signal and becomes Video signal.

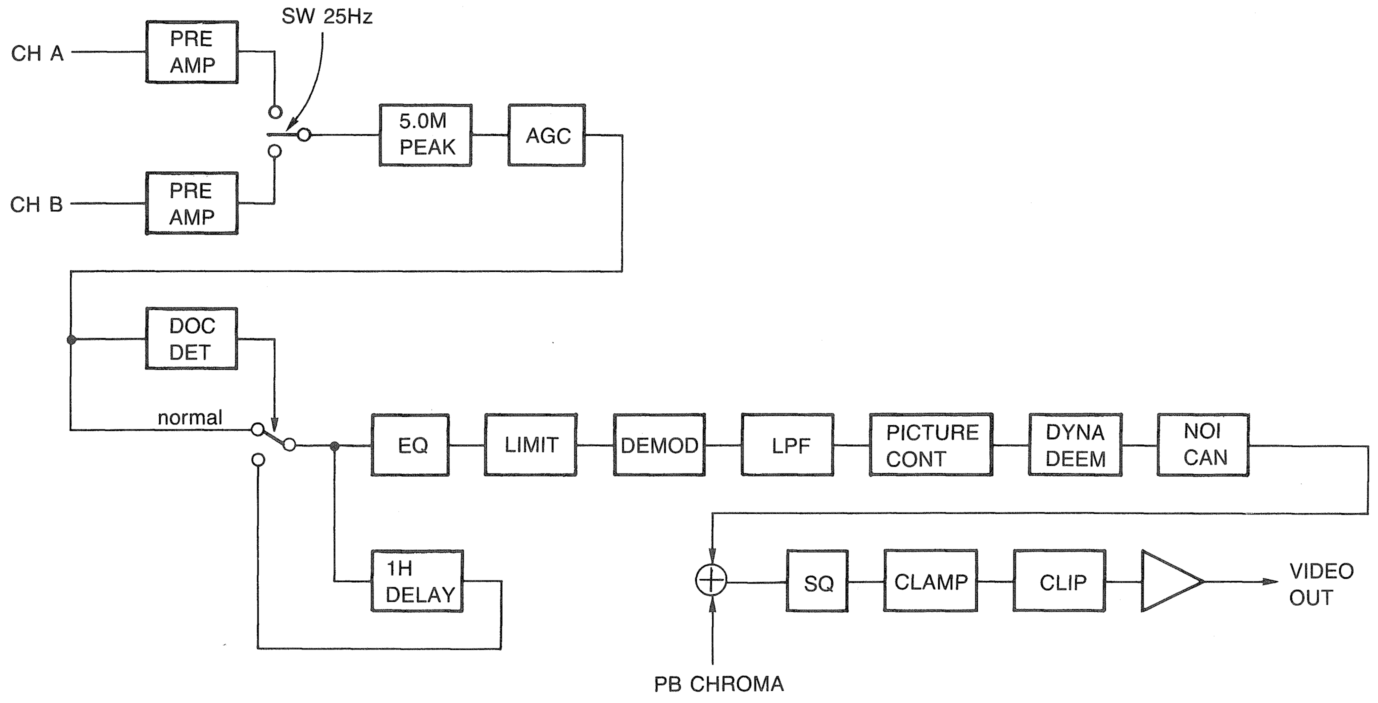


Fig. 9 Block Diagram of Y-signal Playback Circuit

2.3 C-signal Processing in PLAY Mode

C-signal is separated from the reproduced RF signal by 1.5MHz LPF and mixed with 5.06MHz signal by MAIN CONV. 4.43M BPF extracts the 4.43M±500kHz difference signal between the signals of 5.06MHz and 627kHz±500kHz. Then the C-signal in the original frequency range is restored. The signal passes ACC and 2H-delay line to cancel crosstalk and is distributed to PB Amp, APC DET, ACK DET and ACC DET. APC DET compares the phase of the burst signal contained in the C-signal and the reference signal coming from 4.43M VCO. The error signal from APC DET controls 321fH VCO. So 321fH VCO signal contains the jitter component. The 4.43M VCO works as a reference crystal generator in playback while it works as a voltage controlled oscillator in recording. The output of 321fH VCO is divided by 8 as in recording and its frequency is made to 627kHz. The 627kHz signal is mixed with 4.43M reference signal in SUB CONV. Only the mixed signal passes through the 5.06M BPF. As the 627kHz signal contains the input C-signal, the 5.06MHz signal has the same deviation as the input signal. The MAIN CONV puts out the difference signal between the 5.06MHz signal and 627k±500kHz C-signal. Then the output of MAIN CONV is kept at 4.43MHz free from the input frequency fluctuation because the mixed two signals are deviated in the same direction.

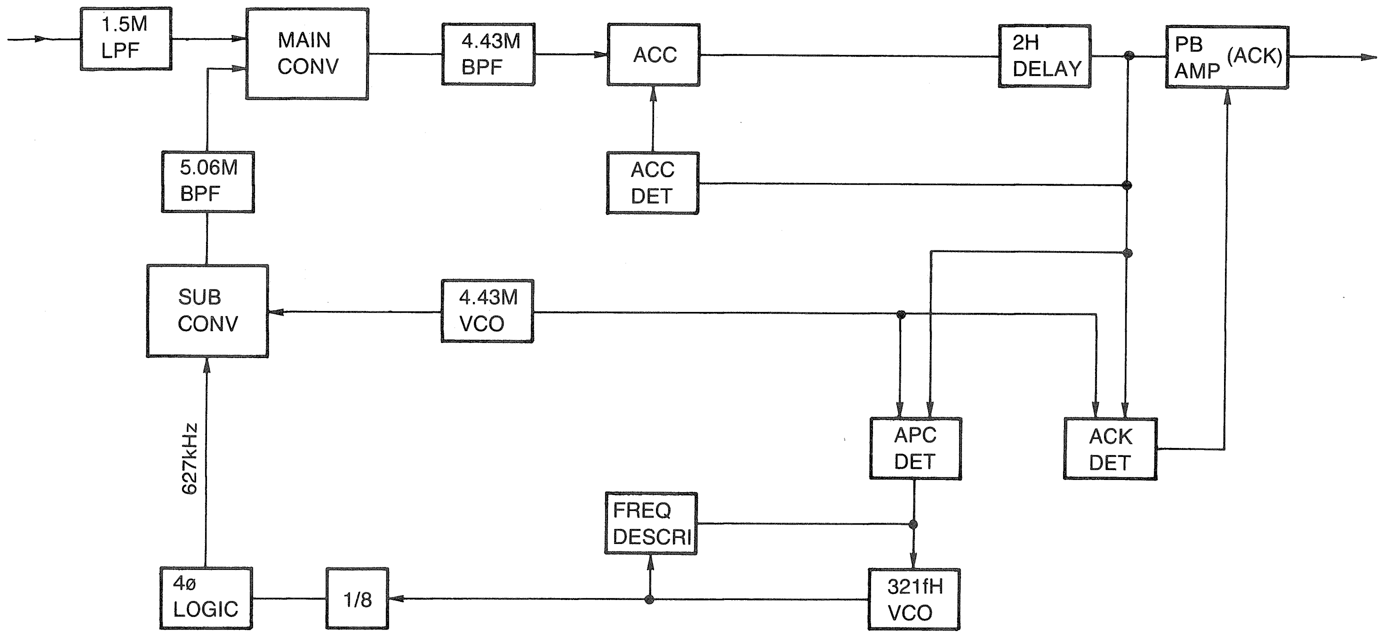


Fig. 10 Block Diagram of C-signal Playback Circuit (VH-600 PAL)

2.4 Servo System

There is no difference in servo system between PAL and NTSC versions. The only difference is in tape speed and the rotational speed of the drum.

Table 3 Specifications of Servo System (VH-600)

Items	PAL	NTSC
Cylinder FG frequency [Hz]	240	200
Servo IC system clock (fc/8) [kHz]	447	554
Capstan FG frequency (SP mode) [Hz]	720	504
Cylinder speed error PWM* [Hz]	1.75k	2.165k
Cylinder phase error PWM [Hz]	1.75k	2.165k
Capstan speed error PWM [Hz]	7k	8.6k
Capstan phase error PWM [Hz]	874	1,082

*PWM: Pulse-width moduration. The duration of a pulse is varied. Its frequency is kept constant.

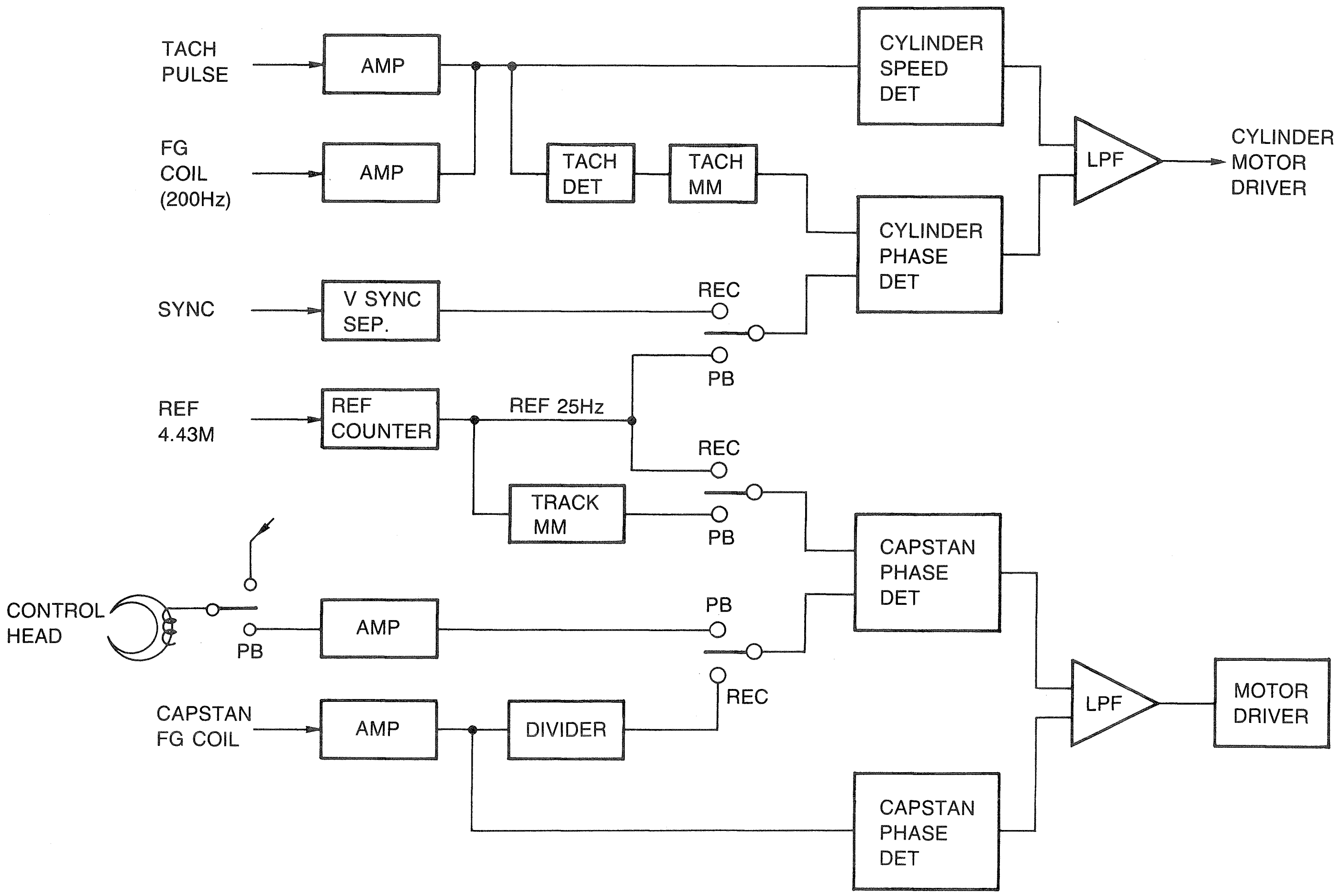


Fig. 11 Block Diagram of Servo System

By H. Kadono

TUNING FORK No. 9

First printing: 1986

Compiled and edited by
Editors in Chief:
Norio Ichimura
General Manager
Product Reliability and
Customers Service Division

Sadahiko Yoshimine
General Manager
Service Department
and
Yasuyuki Yamaoka
Senior Manager
Technical Information and
Coordination Section

Editors:
H. Kishida **T. Kubota**
K. Nohara **H. Kadono**
K. Satoh and **S. Hirose**

PIONEER®

PIONEER ELECTRONIC CORPORATION
4-1, Meguro 1-chome, Meguro-ku, Tokyo 153, Japan

PIONEER ELECTRONICS SERVICE INC.
2265 East 220th Street, Long Beach, California 90810, U.S.A.

PIONEER ELECTRONIC (EUROPE) N.V.
Keetberglaan 1, 2740 Beveren, Belgium

PIONEER ELECTRONICS AUSTRALIA PTY. LTD.
178-184 Boundary Road, Braeside, Victoria 3195, Australia

PIONEER ELECTRONICS OF CANADA, INC.
505 Cochrane Drive, Markham, Ontario L3R 6B8, Canada

Printed in Japan