JUNE. 2001 Rev. 3.2

4-BIT SINGLE CHIP MICROCOMPUTERS

GMS300 SERIES

USER`S MANUAL

- GMS30004
- GMS30012
- GMS30112
- GMS30120
- GMS30140
- GMS30000 EVA

We hereby introduce the manual for CMOS 4-bit microcomputer GMS300 Series.
This manual is prepared for the users who should understand fully the functions and features of GMS300 Series so that you can utilize this product to its fullest capacity. A detailed explanations of the specifications and applications regarding the hardware is hereby provided.

The contents of this user`s manual are subject to change for the reasons of later improvement of the features.
The information, diagrams, and other data in this user`s manual are correct and reliable; however, HYNIX Semiconductor, Inc. is in no way responsible for any violations of patents or other rights of the third party generated by the use of this manual

GMS300 SERIES **Revision History: 1997. APR.**	
Previous Release: 1994.NOV., 1994.JUN., 1994.JAN., 1995.JAN, 1995.AUG., 1996.APR., 1996.JUL.	
Page	Subject(Change since last revision)
Appendix	Modification(Magic IIa Removal)

Table of Contents

Chapter 1

Introduction 1-1

Outline of Characteristics 1-1
Characteristics 1-1
Block Diagram 1-2
Pin Assignment and Dimension 1-3
Electrical Characteristics of GMS300 Series 1-8
I/O circuit types and options 1-10
Masked options 1-13
STOP Function 1-13

Chapter 2

Architecture 2-1
Block Description 2-1

Program Memory (ROM). 2-1
ROM Address Register 2-2
Data Memory (RAM) 2-3
X-Register (X) 2-3
Y-Register (Y) 2-4
Accumulator (Acc) 2-4
Arithmetic and Logic Unit (ALU) 2-4
I/O circuit 2-4
State Counter (SC) 2-5
Clock Generator 2-6
Carrier Generator 2-6
Initial Reset Circuit 2-7
Watch Dog Timer (WDT) 2-7

Chapter 3

Instruction 3-1

Instruction format 3-1
Instruction Table 3-2
Details of Instruction System 3-5
Detailed Description 3-6

Chapter 4

Evaluation Board 4-1

Outline 4-1
Product Specification 4-1
Connection 4-2
Optional Setting 4-3
Caution of Operation 4-6

Chapter 5

Software 5-1

Configuration of Assembler 5-1
Booting up Assembler 5-1
Configuration of Simulator 5-2
Booting up Simulator 5-2
Simulator commands 5-15
Description of commands 5-18
File types used in the simulator 5-48
Error message and troubleshooting 5-49

Appendix

Magic-I A-1
Magic-Ia B-1
Magic-II C-1
Magic-IIb D-1
Magic-IIIa E-1
Magic-IIIb F-1
Magic-IIIc G-1

INTRODUCTION	1
ARCHITECTURE	2
INSTRUCTION	3
EVALUATION BOARD	4
SOFTWARE	5
APPENDIX	6

CHAPTER 1. Introduction

OUTLINE OF CHARACTERISTICS

The GMS300 series is a family of 4-bit, single chip CMOS microcomputer. Since it can form a system by one chip, it contributes to cost reduction and higher efficiency in system.

Characteristics

- Program memory : 512 bytes for GMS30004/012
  1024 bytes for GMS30112/120/140
- Data memory : 32 × 4 bits
- 43 types of instruction set
- 3 levels of subroutine nesting
- 1 bit output port for a large current (REMOUT signal)
- Operating frequency : 300KHz to 1MHz
- Instruction cycle : 12.5usec @480KHz
- CMOS process (Single 3.0V power supply)
- Stop mode (Through internal instruction)
- Released stop mode by key input (Masked option)
- Built in capacitor for ceramic oscillation circuit (Masked option)
- Built in a watch dog timer (WDT)
- Low operating voltage : (2.0~4.0V)

Series	GMS30004	GMS30012	GMS30112	GMS30120	GMS30140
Program memory	512	←	1024	←	←
Data memory	32 × 4	←	←	←	←
I/O ports	-	4	←	←	←
Input ports	4	←	←	←	←
Output ports	6 D0 ~ D5	6 D0 ~ D5	6 D0 ~ D5	8 D0 ~ D7	10 D0 ~ D9
Package	16DIP/SOP	20DIP/SOP	←	24DIP/SOP	←

Table 1-1 GMS300 series members

Block Diagram

The GMS300 series is composed as shown below. These blocks are detailed in CHAPTER2.

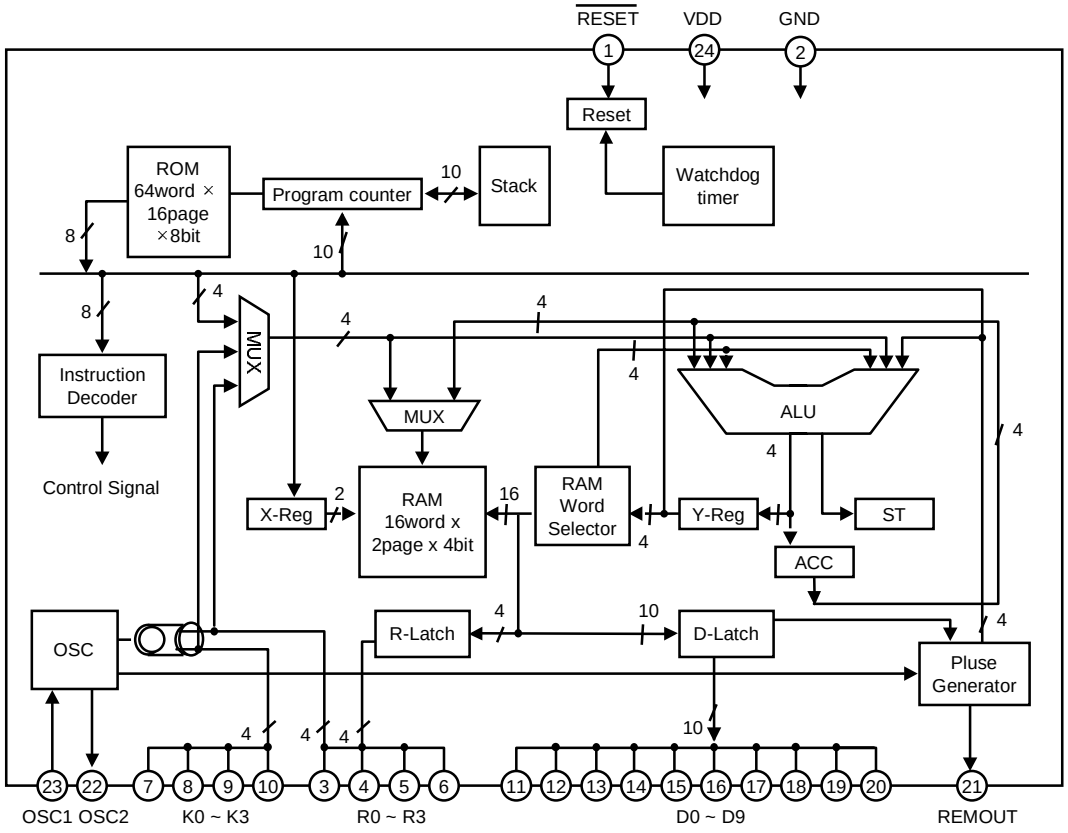


Fig 1-1 Block Diagram (for GMS30140)

Pin Assignment and terminals

Pin Assignment

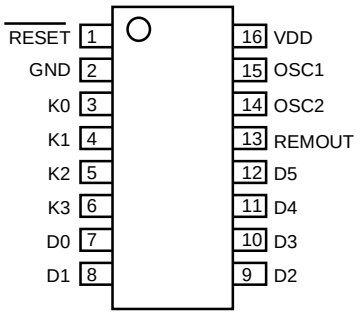


Fig 1-2 GMS30004 Pin Assignment

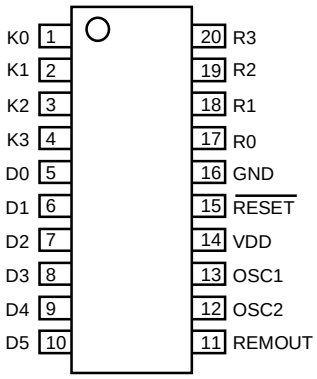


Fig 1-3 GMS30012/112 Pin Assignment

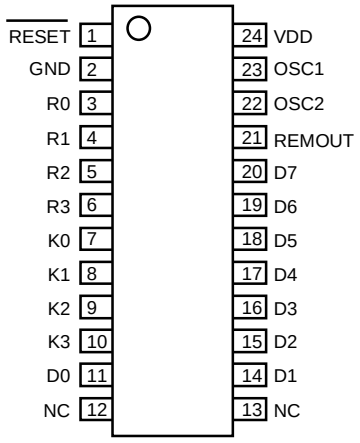


Fig 1-4 GMS30120 Pin Assignment

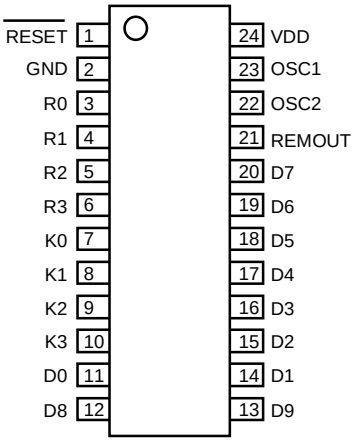


Fig 1-5 GMS30140 Pin Assignment

Pin Dimension

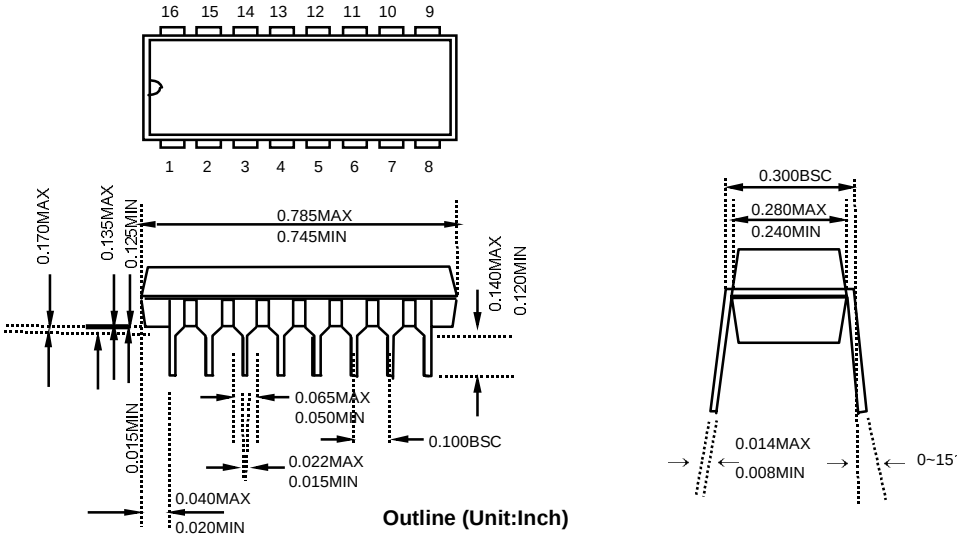


Fig 1-6 16PDIP Pin Dimension

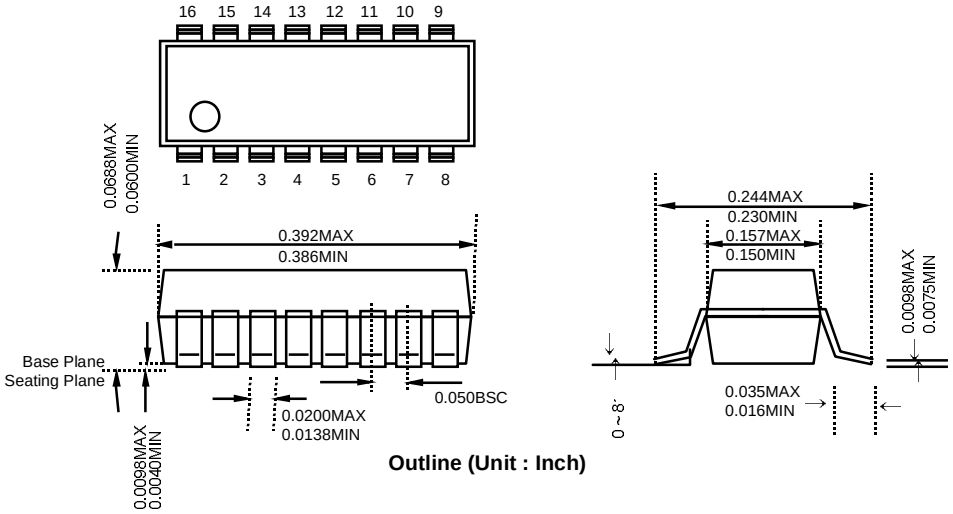
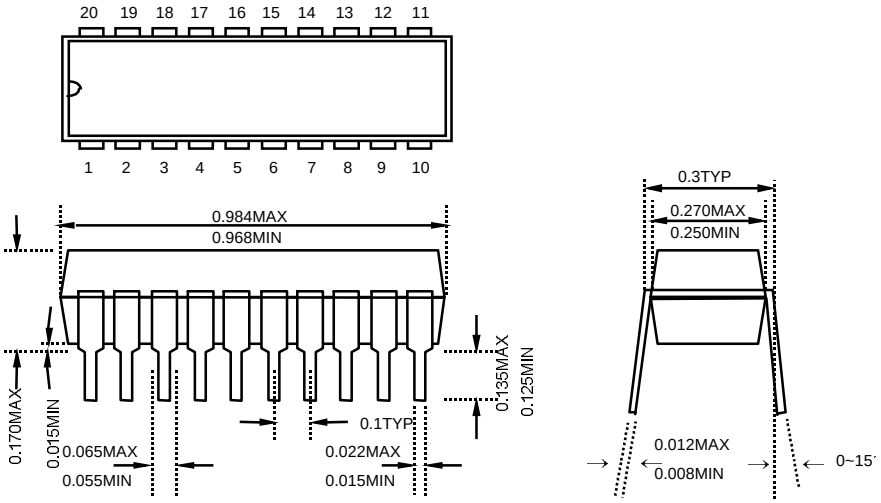
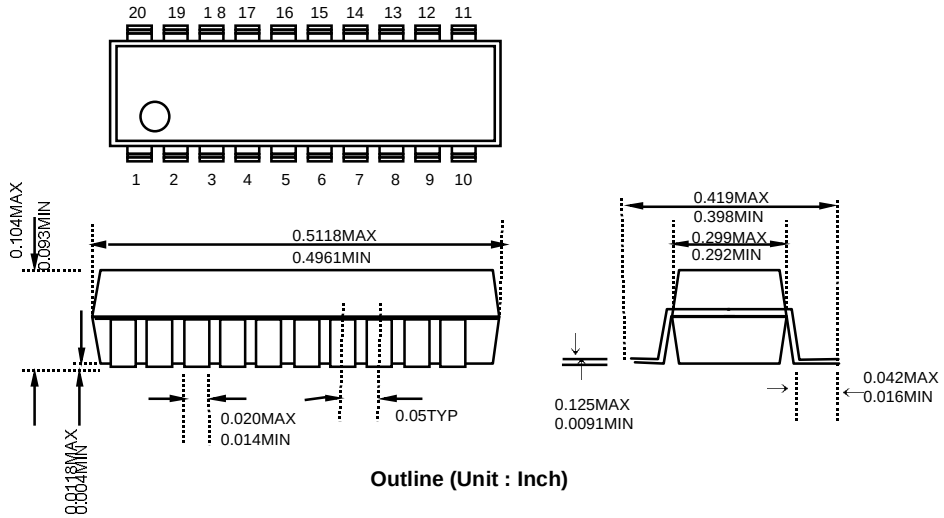


Fig 1-7 16SOP Pin Dimension (150Mil)



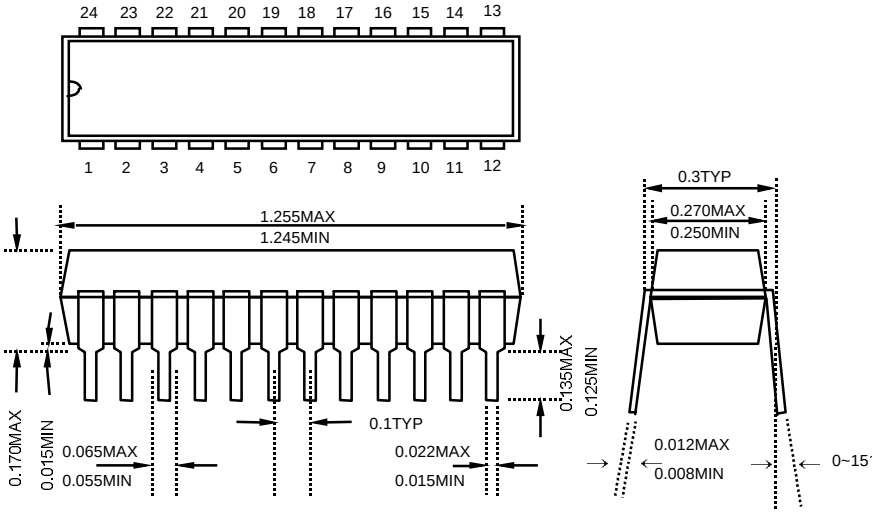
Outline (Unit : Inch)

Fig 1-8 20PDIP Pin Dimension



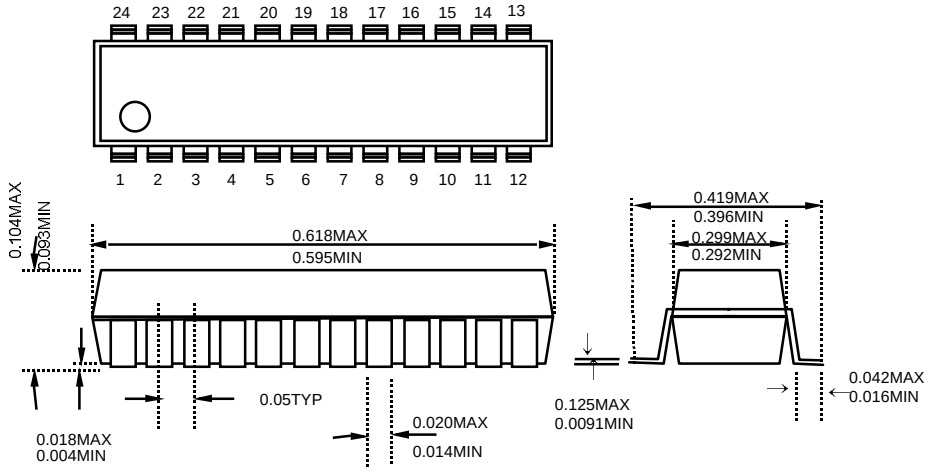
Outline (Unit : Inch)

Fig 1-9 20SOP Pin Dimension



Outline (Unit : Inch)

Fig 1-10 24PDIP Pin Dimension



Outline (Unit : Inch)

Fig 1-11 24SOP Pin Dimension

Pin Description

Pin	Name	Input/Output	Function
VDD			Connected to 2.0~4.0V power supply
GND	Ground		Connected to 0V power supply.
$\overline{\text{RESET}}$	Reset input	I	Reset signal input which is a low active.
OSC1	Oscillator input	I	I/O pins of internal clock oscillating circuit. Built in feedback resistor. Connect a ceramic resonator to these pins.
OSC2	Oscillator output	O	
REMOUT		O	Remocon signal output port which has high current driving capability.
R0 ~ R3	R-Port	I/O	4bit programmable I/O port.
K0 ~ K3	K-Port	I	4bit input port with built in pull-up resistor.
D0 ~ D9	D-Port	O	10bit output port which can be set or reset pin by pin independently. The output structure is N-channel open drain.

Electrical Characteristics for GMS300 series

Absolute maximum ratings (Ta = 25℃)

Parameter	Symbol	Max. rating	Unit
Supply Voltage	V_{DD}	-0.3 ~ 5.0	V
Power dissipation	P_D	700*	mW
Storage temperature range	Tstg	-55 ~ 125	℃
Input voltage	V_{IN}	-0.3 ~ V_{DD}+0.3	V
Output voltage	V_{OUT}	-0.3 ~ V_{DD}+0.3	V

* Thermal derating above 25℃ : 6mW per degree ℃ rise in temperature.

Recommended operation condition

Parameter	Symbol	Rating	Unit
Supply Voltage	V_{DD}	2.2 ~ 4.0	V
Operating temperature	Topr	-20 ~ +70	℃

* In case of using 455KHz resonator.

Electrical characteristics (Ta=25℃, V_{DD}=3V)

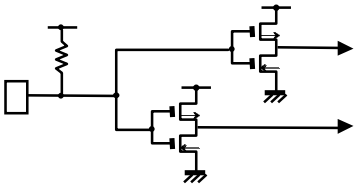
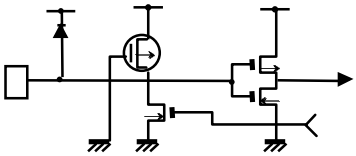
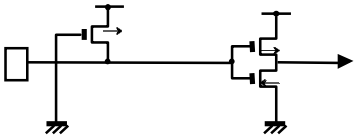
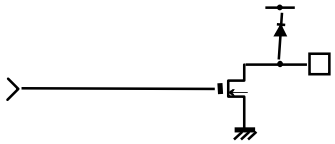
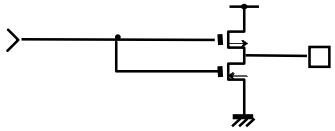
Parameter	Symbol	Limits			Unit	Condition
		Min.	Typ.	Max.		
Input H current	I_{IH}	-	-	1	uA	$V_I=V_{DD}$
$\overline{\text{RESET}}$ input L current	I_{IL2}	-3	-7.5	-16	uA	V_I=GND
K, R input L current	I_{IL1}	-9	-25	-50	uA	V_I=GND, Output off, Pull-up resistor provided.
K, R input H voltage	V_{IH1}	2.1	-	-	V	-
K, R input L voltage	V_{IL1}	-	-	0.9	V	-
$\overline{\text{RESET}}$ input H voltage	V_{IH2}	2.25	-	-	V	-
$\overline{\text{RESET}}$ input L voltage	V_{IL2}	-	-	0.75	V	-
D. R output L voltage	V_{OL2}	-	0.15	0.4	V	I_{OL}=1mA
REMOUT output L voltage	V_{OL1}	-	0.15	0.4	V	I_{OL}=100uA
REMOUT output H voltage	V_{OH1}	2.1	2.5	-	V	I_{OH}=-8mA
OSC2 output L voltage	V_{OL3}	-	0.4	0.9	V	I_{OL}=70uA
OSC2 output H voltage	V_{OH3}	2.1	2.5	-	V	I_{OH}=70uA
D, R output leakage current	I_{OL}	-	-	1	uA	$V_0=V_{DD}$, Output off
Current on STOP mode	I_{STOP}	-	-	1	uA	At STOP mode
Operating supply current 1	I_{DD1}	-	0.3	1.0	mA	f_{OSC} = 455KHz
Operating supply current 2	I_{DD2}	-	0.2 5	-	mA	f_{OSC} = 1MHz
Operating frequency	f_{OSC}	0.3	-	1	MHz	-

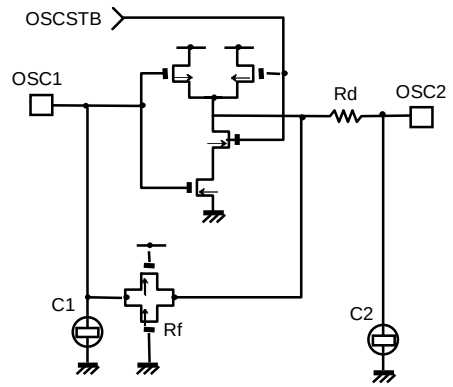
I/O circuit types and options

GMS300 series I/O port types

Pin	I/O	Function
V_{DD}	-	Connected to 2.0~4.0V power supply.
GND	-	Connected to 0V power supply.
$\overline{\text{RESET}}$	Input	Used to input a manual reset. When the pin goes ″L″, the D-output ports and REMOUT-output port are initialized to ″L″, and ROM address is set to address 0 on page 0.
K0~K3	Input	4-bit input port. Released STOP mode built in pull-up resistor by each pin as masked option. (It is released by ″L″ input at STOP)
D0~D9	Output	Each can be set and reset independently. The output is in the form of N-channel-open-drain.
R0~R3	I/O	4-bit I/O port. (Input mode is set only when each of them output ″H″.) In outputting, each can be set and reset independently(or at once.) The output is in the form of N-channel-open-drain. Pull-up resistor and STOP release mode can be respectively selected as masked option for each bit. (It is released by ″L″ input at STOP.)
REMOUT	Output	High current output port. The output is in the form of C-MOS. The state of large current on is ″H″.
OSC1	Input	Oscillator input. Input to the oscillator circuit and connection point for ceramic resonator. Internal capacitors available as masked option. A feedback resistor is connected between this pin and OSC2
OSC2	Output	Connect a ceramic resonator between this pin and OSC1.

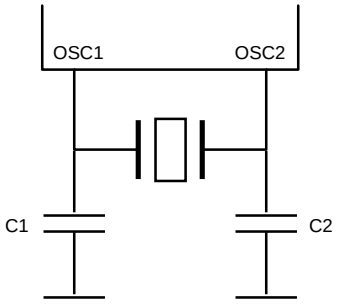
I/O circuit types and options

Pin	I/O	I/O circuit	Note
$\overline{\text{RESET}}$	I		Hysteresis Input Type Built in pull-up-resistor About 400kΩ
R0~R3	I/O		Open drain output ″H″ output at reset (Option) Built in MOS Tr for pull-up About 120kΩ
K0~K3	I		Built in MOS Tr for pull-up About 120kΩ
D0~D9	O		Open drain output ″L″ output at reset
REMOUT	O		CMOS output ″L″ output at reset High current source output

Pin	I/O	I/O circuit	Note
OSC2	O		Built in feedback-Resister About 1㏁ Built in dumping-Resister Rd : About 6㏀
OSC1	I		(Option) Built in resonance Capacitor C1/C2 = 100pF ± n% or C1/C2 = 10 ~ 100pF

◯ : Masked option

***. Recommendable circuit**



Frequency (KHz)	Resonator Maker	Part Name	Load Capacitor	Operating Voltage
320	Murata	CSB320D	C1=C2=220pF	2.0 ~ 4.0V
455	Murata	CSB455E35	C1=C2=Open	2.0 ~ 4.0V
	Kyocera	KBR-455BKTL70	C1=C2=Open	2.0 ~ 4.0V
	Chequers	ZTB-455ET4C	C1=C2=Open	2.0 ~ 4.0V
	TDK	FCR455K3	C1=C2=Open	2.0 ~ 4.0V
480	Murata	CSB480E35	C1=C2=Open	2.0 ~ 4.0V

Masked options

The GMS300 series offer the following optional features.
These options are masked.

1. Watch dog timer reset by REMOUT output signal.
2. Input terminals having STOP release mode : K0~K3, R0~R3.
3. I/O terminals having pull-up resistor : R0~R3
4. Ceramic oscillation circuit contained (or not contained).
5. Output form at stop mode
   D0~D7 : ″L″ or keep before stop mode

STOP Function

Stop mode can be achieved by STOP instructions.
In stop mode :
1. Oscillator is stopped, the operating current is low.
2. Watch dog timer is reset, D8~D9 output and REMOUT output are ″L″.
3. Part other than WDT, D8~D9 output and REMOUT output have a value before come into stop mode.

″But, the state of D0~D7 output in stop mode is able to choose as masked option. ″L″ output or same level before come into stop mode.

The function to release stop mode is able to choose each bit of K or R input.
Stop mode is released when one of K or R input is going to ″L″.

1. State of D0~D7 output and REMOUT output is return to state of before stop mode is achieved.
2. After 1024×8 enable clocks for stable oscillating. First instruction start to operate.
3. In return to normal operation, WDT is counted from zero again.

But, at executing stop instruction, if one of K or R input is chosen to ″L″, stop instruction is same to NOP instruction.

INTRODUCTION	1
ARCHITECTURE	2
INSTRUCTION	3
EVALUATION BOARD	4
SOFTWARE	5
APPENDIX	6

CHAPTER 2. Architecture

BLOCK DESCRIPTION

Characteristics

The GMS300 series can incorporate maximum 1024 words (64 words × 16 pages × 8bits) for program memory. Program counter PC (A_0~A_5) and page address register (A_6~A_9) are used to address the whole area of program memory having an instruction (8bits) to be next executed.
The program memory consists of 64 words on each page, and thus each page can hold up to 64 steps of instructions.
The program memory is composed as shown below.

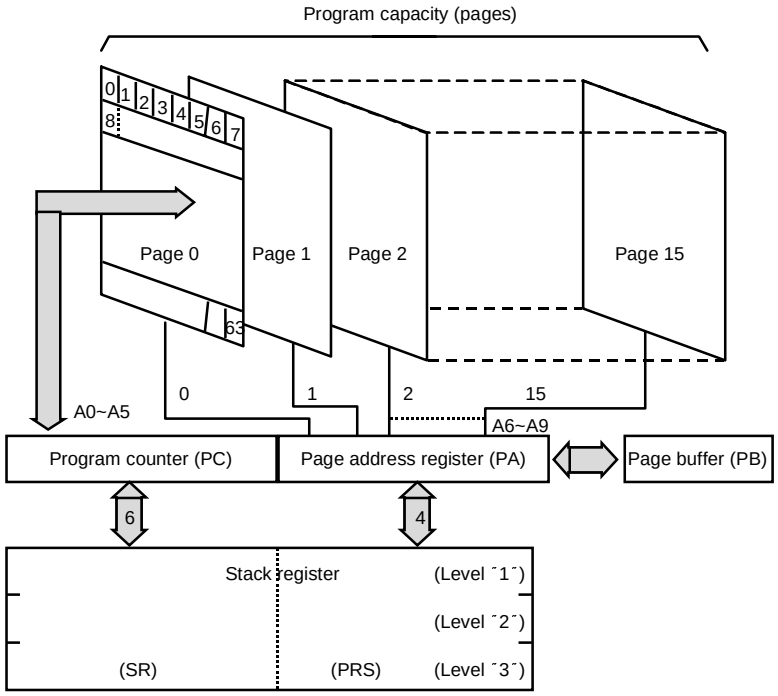


Fig 2-1 Configuration of Program Memory

ROM Address Register

The following registers are used to address the ROM.

- Page address register (PA) :
  Holds ROM`s page number (0~Fh) to be addressed.
- Page buffer register (PB) :
  Value of PB is loaded by an LPBI command when newly addressing a page. Then it is shifted into the PA when rightly executing a branch instruction (BR) and a subroutine call (CAL).
- Program counter (PC) :
  Available for addressing word on each page.
- Stack register (SR) :
  Stores returned-word address in the subroutine call mode.

(1) Page address register and page buffer register :

Address one of pages #0 to #15 in the ROM by the 4-bit binary counter.

Unlike the program counter, the page address register is usually unchanged so that the program will repeat on the same page unless a page changing command is issued. To change the page address, take two steps such as (1) writing in the page buffer what page to jump to (execution of LPBI) and (2) execution of BR or CAL, because and instruction code is of eight bits so that page and word cannot be specified at the same time.
In case a return instruction (RTN) is executed within the subroutine that has been called in the other page, the page address will be changed at the same time.

(2) Program counter :

This 6-bit binary counter increments for each fetch to address a word in the currently addressed page having an instruction to be next executed.
For easier programming, at turning on the power, the program counter is reset to the zero location. The PA is also set to ″0″. Then the program counter specifies the next ROM address in random sequence.
When BR, CAL or RTN instructions are decoded, the switches on each step are turned off not to update the address. Then, for BR or CAL, address data are taken in from the instruction operands (a_0 to a_5), or for RTN, and address is fetched from stack register No. 1.

(3) Stack register :

This stack register provides two stages each for the program counter (6 bits) and the page address register (4bits) so that subroutine nesting can be mode on two levels.

Data memory (RAM)

Up to 32 nibbles (16 words × 2pages × 4bits) is incorporated for storing data. The whole data memory area is indirectly specified by a data pointer (X,Y). Page number is specified by zero bit of X register, and words in the page by 4 bits in Y-register. Data memory is composed in 16 nibbles/page. Figure 2.2 shows the configuration.

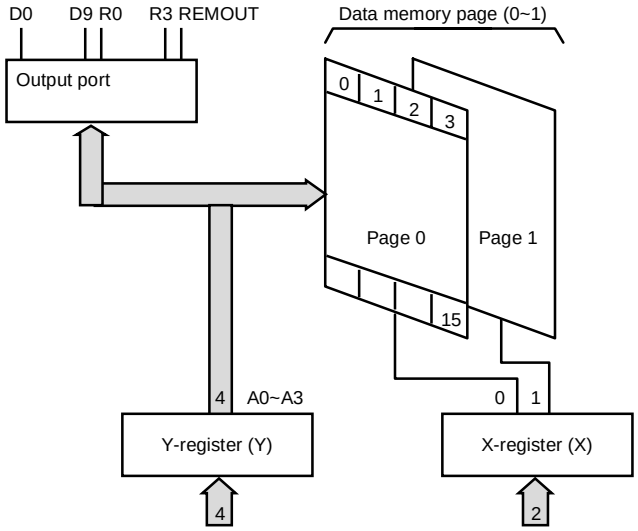


Fig 2-2 Composition of Data Memory

X-register (X)

X-register is consist of 2bit, X0 is a data pointer of page in the RAM, X1 is only used for selecting of D8~D9 with value of Y-register

	X1=0	X1=1
Y=0	D0	D8
Y=1	D1	D9

Table 2-1 Mapping table between X and Y register

Y-register (Y)

Y-register has 4 bits. It operates as a data pointer or a general-purpose register. Y-register specifies and address (a_0~a_3) in a page of data memory, as well as it is used to specify an output port. Further it is used to specify a mode of carrier signal outputted from the REMOUT port. It can also be treated as a general-purpose register on a program.

Accumulator (A_{CC})

The 4-bit register for holding data and calculation results.

Arithmetic and Logic Unit (ALU)

In this unit, 4bits of adder/comparator are connected in parallel as it`s main components and they are combined with status latch and status logic (flag.)

(1) Operation circuit (ALU) :

The adder/comparator serves fundamentally for full addition and data comparison. It executes subtraction by making a complement by processing an inversed output of A_{CC} (A_{CC}+1)

(2) Status logic :

This is to bring an ST, or flag to control the flow of a program. It occurs when a specified instruction is executed in two cases such as overflow in operation and two inputs unequal.

I/O circuit

Ports K0~K3 are 4-bit input ports were pulled up by MOS Tr resistor internally. Ports D0~D9 are output ports, each of which can be independently set and reset. The output is in the form of Nch-open-drain circuit. The input is in the form of a MOS transistor input (MOS Tr resistor pull-up. Masked option). Further, the REMOUT port is a large current driven output port. This port is designed for source current to drive a high current device.

State Counter (SC)

A fundamental machine cycle timing chart is shown below. Every instruction is one byte length. Its execution time is the same. Execution of one instruction takes 6 clocks for fetch cycle and 6 clocks for execute cycle (12 clocks in total). Virtually these two cycles proceed simultaneously, and thus it is apparently completed in 6 clocks (one machine cycle). Exceptionally BR, CAL and RTN instructions is normal execution time since they change an addressing sequencially. Therefore, the next instruction is prefetched so that its execution is completed within the fetch cycle.

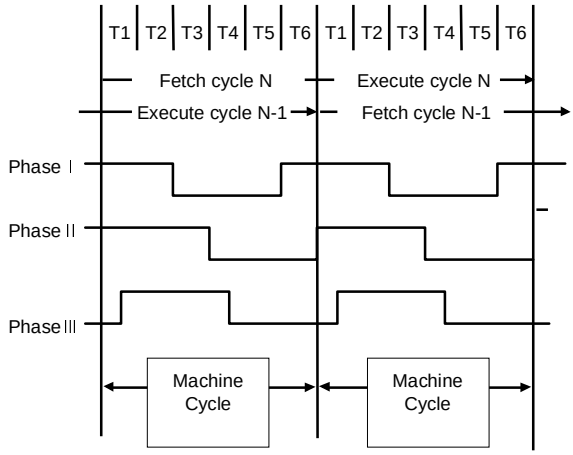
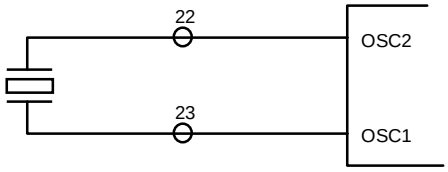


Fig. 2-3 Fundamental timing chart

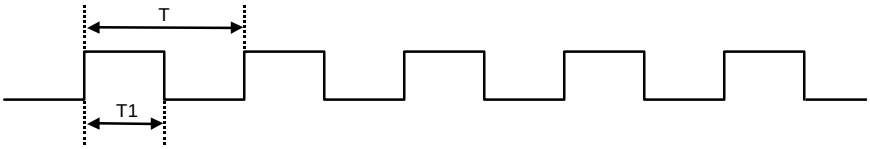
Clock Generator

The GMS300 series has an internal clock oscillator. The oscillator circuit is designed to operate with an external ceramic resonator. Internal capacitors are available as a masked option. Oscillator circuit is able to organize by connecting ceramic resonator to outside. (In order to built in capacitor for oscillation as masked option.)
* It is necessary to connect capacitor to outside in order to change ceramic resonator, You must examine refer to a manufacturer`s



Pulse generator

The following frequency and duty ratio are selected for carrier signal outputted from the REMOUT port depending on a PMR (Pulse Mode Register) value set in a program.



PMR	REMOUT signal
0	$T=1/f_{PUL} = 12/f_{OSC}$, $T1/T = 1/2$
1	$T=1/f_{PUL} = 12/f_{OSC}$, $T1/T = 1/3$
2	$T=1/f_{PUL} = 8/f_{OSC}$, $T1/T = 1/2$
3	$T=1/f_{PUL} = 8/f_{OSC}$, $T1/T = 1/4$
4	$T=1/f_{PUL} = 11/f_{OSC}$, $T1/T = 4/11$
5	No Pulse (same to D0~D9)

* Default value is ″0″

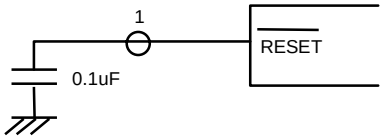
Table 2-2 PMR selection table

Initial Reset Circuit

$\overline{\text{RESET}}$ pin must be down to ″L″ more than 4 machine cycle by outside capacitor or other for power on reset.

The mean of 1 machine cycle is below. 1 machine cycle is $6/f_{OSC}$, however, operating voltage must be in recommended operating conditions, and clock oscillating stability.

* It is required to adjust C value depending on rising time of power supply. (Example shows the case of rising time shorter than 10ms.)



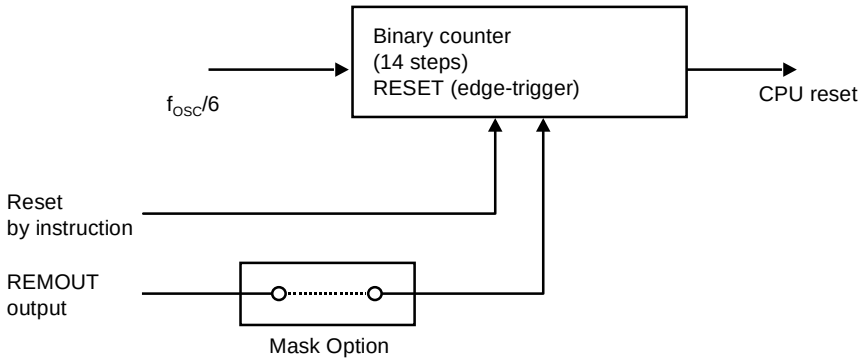
Watch Dog Timer (WDT)

Watch dog timer is organized binary counter of 14 steps. The selected of $f_{OSC}/6$ cycle come in the first step of WDT. If this counter was overflowed, come out reset signal automatically, internal circuit is initialized.

The overflow time is $6 \times 2^{13}/f_{OSC}$ (108.026ms at f_{OSC} = 455KHz).

Normally, the binary counter must be reset before the overflow by using reset instruction (WDTR) or / and REMOUT port (Y-reg=8, SO instruction execution) at masked option.

* It is constantly reset in STOP mode. When STOP is released, counting is restarted. (Refer to 1-13 STOP function>)



INTRODUCTION	1
ARCHITECTURE	2
INSTRUCTION	3
EVALUATION BOARD	4
SOFTWARE	5
APPENDIX	6

CHAPTER 3. Instruction

INSTRUCTION FORMAT

All of the 43 instruction in GMS300 series is format in two fields of OP code and operand which consist of eight bits. The following formats are available with different types of operands.

Format I

All eight bits are for OP code without operand.

Format II

Two bits are for operand and six bits for OP code.
Two bits of operand are used for specifying bits of RAM and X-register (bit 1 and bit 7 are fixed at ˝0˝)

Format III

Four bits are for operand and the others are OP code.
Four bits of operand are used for specifying a constant loaded in RAM or Y-register, a comparison value of compare command, or page addressing in ROM.

Format IV

Six bits are for operand and the others are OP code.
Six bits of operand are used for word addressing in the ROM.

INSTRUCTION TABLE

The GMS300 series provides the following 43 basic instructions.

	Category	Mnemonic	Function	ST*1
1	Register to Register	LAY	A ← Y	S
2		LYA	Y ← A	S
3		LAZ	A ← 0	S
4	RAM to Register	LMA	M(X,Y) ← A	S
5		LMAIY	M(X,Y) ← A, Y ← Y+1	S
6		LYM	Y ← M(X,Y)	S
7		LAM	A ← M(X,Y)	S
8		XMA	A ↔ M(X,Y)	S
9	Immediate	LYI i	Y ← i	S
10		LMIIY i	M(X,Y) ← i, Y ← Y+1	S
11		LXI n	X ← n	S
12	RAM Bit Manipulation	SEM n	M(n) ← 1	S
13		REM n	M(n) ← 0	S
14		TM n	TEST M(n) = 1	E
15	ROM Address	BR a	if ST = 1 then Branch	S
16		CAL a	if ST = 1 then Subroutine call	S
17		RTN	Return from Subroutine	S
18		LPBI i	PB ← i	S
19	Arithmetic	AM	A ← A + M(X,Y)	C
20		SM	A ← M(X,Y) - A	B
21		IM	A ← M(X,Y) + 1	C
22		DM	A ← M(X,Y) - 1	B
23		IA	A ← A + 1	S
24		IY	Y ← Y + 1	C
25		DA	A ← A - 1	B

	Category	Mnemonic	Function	ST*1
26	Arithmetic	DY	Y ← Y - 1	B
27		EORM	A ← A ⊕ M (X,Y)	S
28		NEGA	A ← $\overline{A}$ + 1	Z
29	Comparison	ALEM	TEST A ≤ M(X,Y)	E
30		ALEI i	TEST A ≤ i	E
31		MNEZ	TEST M(X,Y) ≠ 0	N
32		YNEA	TEST Y ≠ A	N
33		YNEI i	TEST Y ≠ i	N
34		KNEZ	TEST K ≠ 0	N
35		RNEZ	TEST R ≠ 0	N
36	Input / Output	LAK	A ← K	S
37		LAR	A ← R	S
38		SO	Output(Y) ← 1*2	S
39		RO	Output(Y) ← 0*2	S
40	Control	WDTR	Watch Dog Timer Reset	S
41		STOP	Stop operation	S
42		LPY	PMR ← Y	S
43		NOP	No operation	S

Note) i = 0~f, n = 0~3, a = 6bit PC Address

*1 Column ST indicates conditions for changing status. Symbols have the following meanings

S : On executing an instruction, status is unconditionally set.
C : Status is only set when carry or borrow has occurred in operation.
B : Status is only set when borrow has not occurred in operation.
E : Status is only set when equality is found in comparison.
N : Status is only set when equality is not found in comparison.
Z : Status is only set when the result is zero.

*2 Operation is settled by a value of Y-register.

Value of X-reg	Value of Y-reg	Operation
0 or 1	0~7	SO : D(Y) ← 1, RO : D(Y) ← 0
0 or 1	8	REMOUT port repeats ″H″ and ″L″ in pulse frequency. (when PMR = 5, it is fixed at ″H″) SO : REMOUT (PMR) ← 1 RO : REMOUT (PMR) ← 0
0 or 1	9	SO : D0 ~ D9 ← 1 (High-Z) R0 : D0 ~ D9 ← 0
0 or 1	A ~ D	SO : R(Y-Ah) ← 1, RO : R(Y-Ah) ← 0
0 or 1	E	SO : R0 ~ R3 ← 1, RO : R0~R3 ← 0
0 or 1	F	SO : D0 ~ D9 ← 1 (High-Z) R0~R3 ← 1 R0 : D0 ~ D9 ← 0 R0~R3 ← 0
2 or 3	0	SO : D(8) ← 1, RO : D(8) ← 0
2 or 3	1	SO : D(9) ← 1, RO : D(9) ← 0

DETAILS OF INSTRUCTION SYSTEM

All 43 basic instructions of the GMS300 Series are one by one described in detail below.

Description Form

Each instruction is headlined with its mnemonic symbol according to the instructions table given earlier.
Then, for quick reference, it is described with basic items as shown below. After that, detailed comment follows.

- Items :

  - Naming : Full spelling of mnemonic symbol
  - Status : Check of status function
  - Format : Categorized into Ⅰ to Ⅳ
  - Operand : Omitted for Format Ⅰ
  - Function

Detailed Description

(1) LAY

Naming :	Load Accumulator from Y-Register
Status :	Set
Format :	I
Function :	A ← Y
<Comment>	Data of four bits in the Y-register is unconditionally transferred to the accumulator. Data in the Y-register is left unchanged.

(2) LYA

Naming :	Load Y-register from Accumulator
Status :	Set
Format :	I
Function :	Y ← A
<Comment>	Load Y-register from Accumulator

(3) LAZ

Naming :	Clear Accumulator
Status :	Set
Format :	I
Function :	A ← 0
<Comment>	Data in the accumulator is unconditionally reset to zero.

(4) LMA

Naming :	Load Memory from Accumulator
Status :	Set
Format :	I
Function :	M(X,Y) ← A
<Comment>	Data of four bits from the accumulator is stored in the RAM location addressed by the X-register and Y-register. Such data is left unchanged.

(5) LMAIY

Naming :	Load Memory from Accumulator and Increment Y-Register
Status :	Set
Format :	I
Function :	M(X,Y) ← A, Y ← Y+1
<Comment>	Data of four bits from the accumulator is stored in the RAM location addressed by the X-register and Y-register. Such data is left unchanged.

(6) LYM

Naming :	Load Y-Register form Memory
Status :	Set
Format :	I
Function :	Y ← M(X,Y)
<Comment>	Data from the RAM location addressed by the X-register and Y-register is loaded into the Y-register. Data in the memory is left unchanged.

(7) LAM

Naming :	Load Accumulator from Memory
Status :	Set
Format :	I
Function :	A ← M(X,Y)
<Comment>	Data from the RAM location addressed by the X-register and Y-register is loaded into the Y-register. Data in the memory is left unchanged.

(8) XMA

Naming :	Exchanged Memory and Accumulator
Status :	Set
Format :	I
Function :	M(X,Y) ↔ A
<Comment>	Data from the memory addressed by X-register and Y-register is exchanged with data from the accumulator. For example, this instruction is useful to fetch a memory word into the accumulator for operation and store current data from the accumulator into the RAM. The accumulator can be restored by another XMA instruction.

(9) LYI i

Naming :	Load Y-Register from Immediate
Status :	Set
Format :	III
Operand :	Constant $0 \leq i \leq 15$
Function :	Y ← i
<Purpose>	To load a constant in Y-register. It is typically used to specify Y-register in a particular RAM word address, to specify the address of a selected output line, to set Y-register for specifying a carrier signal outputted from OUT port, and to initialize Y-register for loop control. The accumulator can be restored by another XMA instruction.
<Comment>	Data of four bits from operand of instruction is transferred to the Y-register.

(10) LMIIY i

Naming :	Load Memory from Immediate and Increment Y-Register
Status :	Set
Format :	III
Operand :	Constant $0 \leq i \leq 15$
Function :	$M(X,Y) \leftarrow i, \quad Y \leftarrow Y + 1$
<Comment>	Data of four bits from operand of instruction is stored into the RAM location addressed by the X-register and Y-register. Then data in the Y-register is incremented by one.

(11) LXI n

Naming :	Load X-Register from Immediate
Status :	Set
Format :	II
Operand :	X file address $0 \leq n \leq 3$
Function :	$X \leftarrow n$
<Comment>	A constant is loaded in X-register. It is used to set X-register in an index of desired RAM page. Operand of 1 bit of command is loaded in X-register.

(12) SEM n

Naming :	Set Memory Bit
Status :	Set
Format :	II
Operand :	Bit address $0 \leq n \leq 3$
Function :	$M(X,Y,n) \leftarrow 1$
<Comment>	Depending on the selection in operand of operand, one of four bits is set as logic 1 in the RAM memory addressed in accordance with the data of the X-register and Y-register.

(13) REM n

Naming :	Reset Memory Bit
Status :	Set
Format :	II
Operand :	Bit address $0 \leq n \leq 3$
Function :	$M(X,Y,n) \leftarrow 0$
<Comment>	Depending on the selection in operand of operand, one of four bits is set as logic 0 in the RAM memory addressed in accordance with the data of the X-register and Y-register.

(14) TM n

Naming :	Test Memory Bit
Status :	Comparison results to status
Format :	II
Operand :	Bit address $0 \leq n \leq 3$
Function :	M(X,Y,n) $\leftarrow$ 1? ST $\leftarrow$ 1 when M(X,Y,n)=1, ST $\leftarrow$ 0 when M(X,Y,n)=0
<Purpose>	A test is made to find if the selected memory bit is logic. 1 Status is set depending on the result.

(15) BR a

Naming :	Branch on status 1
Status :	Conditional depending on the status
Format :	IV
Operand :	Branch address a (Addr)
Function :	When ST =1 , PA $\leftarrow$ PB, PC $\leftarrow$ a(Addr) When ST = 0, PC $\leftarrow$ PC + 1, ST $\leftarrow$ 1 Note : PC indicates the next address in a fixed sequence that is actually pseudo-random count.
<Purpose>	For some programs, normal sequential program execution can be change. A branch is conditionally implemented depending on the status of results obtained by executing the previous instruction.
<Comment>	• Branch instruction is always conditional depending on the status. a. If the status is reset (logic 0), a branch instruction is not rightly executed but the next instruction of the sequence is executed. b. If the status is set (logic 1), a branch instruction is executed as follows. • Branch is available in two types - short and long. The former is for addressing in the current page and the latter for addressing in the other page. Which type of branch to exeute is decided according to the PB register. To execute a long branch, data of the PB register should in advance be modified to a desired page address through the LPBI instruction.

(16) CAL a

Naming : Subroutine Call on status 1
Status : Conditional depending on the status
Format : IV
Operand : Subroutine code address a(Addr)
Function : When ST =1 , PC ← a(Addr) PA ← PB
SR1 ← PC + 1, PSR1 ← PA
SR2 ← SR1 PSR2 ← PSR1
SR3 ← SR2 PSR3 ← PSR2
When ST = 0 PC ← PC + 1 PB ← PS ST ← 1
Note : PC actually has pseudo-random count against the next instruction.

<Comment>

- In a program, control is allowed to be transferred to a mutual subroutine. Since a call instruction preserves the return address, it is possible to call the subroutine from different locations in a program, and the subroutine can return control accurately to the address that is preserved by the use of the call return instruction (RTN).
  Such calling is always conditional depending on the status.

  a. If the status is reset, call is not executed.
  b. If the status is set, call is rightly executed.

The subroutine stack (SR) of three levels enables a subroutine to be manipulated on three levels. Besides, a long call (to call another page) can be executed on any level.

- For a long call, an LPBI instruction should be executed before the CAL. When LPBI is omitted (and when PA=PB), a short call (calling in the same page) is executed.

(17) RTN

Naming : Return from Subroutine
Status : Set
Format : |
Function :

PC ← SR1	PA, PB ← PSR1
SR1 ← SR2	PSR1 ← PSR2
SR2 ← SR3	PSR2 ← PSR3
SR3 ← SR3	PSR3 ← PSR2
	ST ← 1

<Purpose> Control is returned from the called subroutine to the calling program.
<Comment> Control is returned to its home routine by transferring to the PC the data of the return address that has been saved in the stack register (SR1).
At the same time, data of the page stack register (PSR1) is transferred to the PA and PB.

(18) LPBI i

Naming : Load Page Buffer Register from Immediate
Status : Set
Format : III
Operand : ROM page address $0 \leq i \leq 15$
Function : PB ← i
<Purpose> A new ROM page address is loaded into the page buffer register (PB).
This loading is necessary for a long branch or call instruction.
<Comment> The PB register is loaded together with three bits from 4 bit operand.

(19) AM

Naming : Add Accumulator to Memory and Status 1 on Carry
Status : Carry to status
Format : |
Function : A ← M(X,Y)+A, ST ← 1(when total>15),
ST ← 0 (when total ≤15)
<Comment> Data in the memory location addressed by the X and Y-register is added to data of the accumulator. Results are stored in the accumulator. Carry data as results is transferred to status. When the total is more than 15, a carry is caused to put ″1″ in the status. Data in the memory is not changed.

(20) SM

Naming : Subtract Accumulator to Memory and Status 1 Not Borrow
Status : Carry to status
Format : |
Function : A ← M(X,Y) - A ST ← 1(when A ≤ M(X,Y))
ST ← 0(when A > M(X,Y))
<Comment> Data of the accumulator is, through a 2`s complemental addition, subtracted from the memory word addressed by the Y-register. Results are stored in the accumulator. If data of the accumulator is less than or equal to the memory word, the status is set to indicate that a borrow is not caused.
If more than the memory word, a borrow occurs to reset the status to ″0″.

(21) IM

Naming : Increment Memory and Status 1 on Carry
Status : Carry to status
Format : |
Function : A ← M(X,Y) + 1 ST ← 1(when M(X,Y) ≥ 15)
ST ← 0(when M(X,Y) < 15)
<Comment> Data of the memory addressed by the X and Y-register is fetched. Adding 1 to this word, results are stored in the accumulator. Carry data as results is transferred to the status. When the total is more than 15, the status is set. The memory is left unchanged.

(22) DM

Naming : Decrement Memory and Status 1 on Not Borrow
Status : Carry to status
Format : |
Function : A ← M(X,Y) - 1 ST ← 1(when M(X,Y) ≥1)
ST ← 0 (when M(X,Y) = 0)
<Comment> Data of the memory addressed by the X and Y-register is fetched, and one is subtracted from this word (addition of Fh)> Results are stored in the accumulator. Carry data as results is transferred to the status. If the data is more than or equal to one, the status is set to indicate that no borrow is caused. The memory is left unchanged.

(23) IA

Naming : Increment Accumulator
Status : Set
Format : |
Function : A ← A+1
<Comment> Data of the accumulator is incremented by one. Results are returned to the accumulator.
A carry is not allowed to have effect upon the status.

(24) IY

Naming : Increment Y-Register and Status 1 on Carry
Status : Carry to status
Format : |
Function : Y ← Y + 1 ST ← 1 (when Y = 15)
ST ← 0 (when Y < 15)
<Comment> Data of the Y-register is incremented by one and results are returned to the Y-register.
Carry data as results is transferred to the status. When the total is more than 15, the status is set.

(25) DA

Naming : Decrement Accumulator and Status 1 on Borrow
Status : Carry to status
Format : |
Function : A ← A - 1 ST ← 1(when A ≥1)
ST ← 0 (when A = 0)
<Comment> Data of the accumulator is decremented by one. As a result (by addition of Fh), if a borrow is caused, the status is reset to ″0″ by logic. If the data is more than one, no borrow occurs and thus the status is set to ″1″.

(26) DY

Naming : Decrement Y-Register and Status 1 on Not Borrow
Status : Carry to status
Format : |
Function : Y ← Y -1 ST ← 1 (when Y ≥ 1)
ST ← 0 (when Y = 0)
<Purpose> Data of the Y-register is decremented by one.
<Comment> Data of the Y-register is decremented by one by addition of minus 1 (Fh).
Carry data as results is transferred to the status. When the results is equal to 15, the status is set to indicate that no borrow has not occurred.

(27) EORM

Naming : Exclusive or Memory and Accumulator
Status : Set
Format : |
Function : A ← M(X,Y) ⊕ A
<Comment> Data of the accumulator is, through a Exclusive OR, subtracted from the memory word addressed by X and Y-register. Results are stored into the accumulator.

(28) NEGA

Naming : Negate Accumulator and Status 1 on Zero
Status : Carry to status
Format : |
Function : A ← $\overline{A}$ + 1 ST ← 1(when A = 0)
ST ← 0 (when A != 0)
<Purpose> The 2`s complement of a word in the accumulator is obtained.
<Comment> The 2`s complement in the accumulator is calculated by adding one to the 1`s complement in the accumulator. Results are stored into the accumulator. Carry data is transferred to the status. When data of the accumulator is zero, a carry is caused to set the status to ″1″.

(29) ALEM

Naming : Accumulator Less Equal Memory
Status : Carry to status
Format : |
Function : A ≤ M(X,Y) ST ← 1 (when A ≤ M(X,Y))
ST ← 0 (when A > M(X,Y))
<Comment> Data of the accumulator is, through a complemental addition, subtracted from data in the memory location addressed by the X and Y-register. Carry data obtained is transferred to the status. When the status is ″1″, it indicates that the data of the accumulator is less than or equal to the data of the memory word. Neither of those data is not changed.

(30) ALEI

Naming : Accumulator Less Equal Immediate
Status : Carry to status
Format : III
Function : A ≤ i ST ← 1 (when A ≤ i)
ST ← 0 (when A > i)
<Purpose> Data of the accumulator and the constant are arithmetically compared.
<Comment> Data of the accumulator is, through a complemental addition, subtracted from the constant that exists in 4bit operand. Carry data obtained is transferred to the status. The status is set when the accumulator value is less than or equal to the constant. Data of the accumulator is left unchanged.

(31) MNEZ

Naming : Memory Not Equal Zero
Status : Comparison results to status
Format : |
Function : M(X,Y) ≠ 0 ST ← 1(when M(X,Y) ≠ 0)
ST ← 0 (when M(X,Y) = 0)
<Purpose> A memory word is compared with zero.
<Comment> Data in the memory addressed by the X and Y-register is logically compared with zero. Comparison data is thransferred to the status. Unless it is zero, the status is set.

(32) YNEA

Naming : Y-Register Not Equal Accumulator
Status : Comparison results to status
Format : |
Function : Y ≠ A ST ← 1 (when Y ≠ A)
ST ← 0 (when Y = A)
<Purpose> Data of Y-register and accumulator are compared to check if they are not equal.
<Comment> Data of the Y-register and accumulator are logically compared. Results are transferred to the status. Unless they are equal, the status is set.

(33) YNEI

Naming : Y-Register Not Equal Immediate
Status : Comparison results to status
Format : III
Operand : Constant 0 ≤ i ≤ 15
Function : Y ≠ i ST ← 1 (when Y ≠ i)
ST ← 0 (when Y = i)
<Comment> The constant of the Y-register is logically compared with 4bit operand. Results are transferred to the status. Unless the operand is equal to the constant, the status is set.

(34) KNEZ

Naming : K Not Equal Zero
Status : The status is set only when not equal
Format : |
Function : When K ≠ 0, ST ← 1
<Purpose> A test is made to check if K is not zero.
<Comment> Data on K are compared with zero. Results are transferred to the status. For input data not equal to zero, the status is set.

(35) RNEZ

Naming : R Not Equal Zero
Status : The status is set only when not equal
Format : |
Function : When R ≠ 0, ST ← 1
<Purpose> A test is made to check if R is not zero.
<Comment> Data on R are compared with zero. Results are transferred to the status. For input data not equal to zero, the status is set.

(36) LAK

Naming : Load Accumulator from K
Status : Set
Format : |
Function : A ← K
<Comment> Data on K are transferred to the accumulator

(37) LAR

Naming : Load Accumulator from R
Status : Set
Format : |
Function : A ← R
<Comment> Data on R are transferred to the accumulator

(38) SO

Naming : Set Output Register Latch
Status : Set
Format : |
Function :

D(Y) ← 1	0 ≤ Y ≤ 7
REMOUT ← 1(PMR=5)	Y = 8
D0~D9 ← 1 (High-Z)	Y = 9
R(Y) ← 1	Ah ≤ Y ≤ Dh
R ← 1	Y = Eh
D0~D9, R ← 1	Y = Fh

<Purpose> A single D output line is set to logic 1, if data of Y-register is between 0 to 7.
Carrier frequency come out from REMOUT port, if data of Y-register is 8.
All D output line is set to logic 1, if data of Y-register is 9.
It is no operation, if data of Y-register between 10 to 15.
When Y is between Ah and Dh, one of R output lines is set at logic 1.
When Y is Eh, the output of R is set at logic 1.
When Y is Fh, the output D0~D9 and R are set at logic 1.

<Comment> Data of Y-register is between 0 to 7, selects appropriate D output.
Data of Y-register is 8, selects REMOUT port.
Data of Y-register is 9, selects all D port.
Data in Y-register, when between Ah and Dh, selects an appropriate R output (R0~R3).
Data in Y-register, when it is Eh, selects all of R0~R3.
Data in Y-register, when it is Fh, selects all of D0~D9 and R0~R3.

(39) RO

Naming : Reset Output Register Latch
Status : Set
Format : |
Function :

D(Y) ← 0	0 ≤ Y ≤ 7
REMOUT ← 0	Y = 8
D0~D9 ← 0	Y = 9
R(Y) ← 0	Ah ≤ Y ≤ Dh
R ← 0	Y = Eh
D0~D9, R ← 0	Y = Fh

<Purpose> A single D output line is set to logic 0, if data of Y-register is between 0 to 9.
REMOUT port is set to logic 0, if data of Y-register is 9.
All D output line is set to logic 0, if data of Y-register is 9.
When Y is between Ah and Dh, one of R output lines is set at logic 0.
When Y is Eh, the output of R is set at logic 0
When Y is Fh, the output D0~D9 and R are set at logic 1.

<Comment> Data of Y-register is between 0 to 7, selects appropriate D output.
Data of Y-register is 8, selects REMOUT port.
Data of Y-register is 9, selects D port.
Data in Y-register, when between Ah and Dh, selects an appropriate R output (R0~R3).
Data in Y-register, when it is Eh, selects all of R0~R3.
Data in Y-register, when it is Fh, selects all of D0~D9 and R0~R3.

(40) WDTR

Naming : Watch Dog Timer Reset
Status : Set
Format : |
Function : Reset Watch Dog Timer (WDT)
<Purpose> Normally, you should reset this counter before overflowed counter for dc watch dog timer. this instruction controls this reset signal.

(41) STOP

Naming : STOP
Status : Set
Format : |
Function : Operate the stop function
<Purpose> Stopped oscillator, and little current.
(See 1-12 page, STOP function.)

(42) LPY

Naming : Pulse Mode Set
Status : Set
Format : |
Function : PMR ← Y
<Comment> Selects a pulse signal outputted from REMOUT port.

(43) NOP

Naming : No Operation
Status : Set
Format : |
Function : No operation

INTRODUCTION	1
ARCHITECTURE	2
INSTRUCTION	3
EVALUATION BOARD	4
SOFTWARE	5
APPENDIX	6

CHAPTER 4. Evaluation Board

OUTLINE

The GMS 30000 EVA is an evaluation board for GMS300 series, 4-bit, 1-chip microcomputer. It is designed to evaluate and confirm the operations of the application system in the nearest possible form of final products while it is under development.

The major features are as follows :

- The GMS 30000 EVA is used for the evaluation chip.
- The board is connected to the application system through an connection cable (DIP24).
- EPROM of 2764, 27128, and 27256 are used for the program memory.
- The instruction system and I/O specifications are basically the same as those of the GMS300 series.

Product Specifications

• GMS 30000 EVA board module	Dimensions	64 × 82 (mm)
	Supply Voltage	2.5 ~ 5.5 (V)
	Operating temperature	0~50 (℃)
• Connection cable	DIP 24 cable	

Connection

Perform emulation with the following connectors.

[User] Connection socket

The cable for the target system is connected.

Pin No.	Signal	Pin No.	Signal
1	$\overline{\text{Reset}}$	13	D9
2	GND	14	D1
3	R0	15	D2
4	R1	16	D3
5	R2	17	D4
6	R3	18	D5
7	K0	19	D6
8	K1	20	D7
9	K2	21	Remout
10	K3	22	OSC2
11	D0	23	OSC1
12	D8	24	VDD

[M1] Monitor pin

Operations inside the GMS 30000 EVA can be monitored. Signals that can be monitored are as follows.

AC0~AC3, X0, X1, Y0~Y3, REMDATA, CK2, CK5, WDTR, GND

[M2] Oscillation monitoring pin

The oscillation output signal can be monitored.

[T1] D8 output monitoring pin

The D8 output signal can be monitored.

[T2] D9 output monitoring pin

The D9 output signal can be monitored.

Optional setting

The following optional setting in accordance with the application system specifications is required :

[S1] Optional mask setting

Optional masks available with GMS300 series units can be set by selecting short posts.

1. Setting of K-input and R-Port for STOP release
   Shorting the KSR0 ~ KSR3 and RSR0 ~ RSR3 with the side of H can set the STOP releasing function by the corresponding KSR0 ~ KSR3 and RSR0 ~ RSR3. If no STOP releasing function is desired, short them with the side of L.

Setting pin	K0	K1	K2	K3	R0	R1	R2	R3
Short post	KSR0	KSR1	KSR2	KSR3	RSR0	RSR1	RSR2	RSR3
Setting of STOP	H	H	H	H	H	H	H	H
No setting of STOP	L	L	L	L	L	L	L	L

2. Setting of pull-up resistor built-in R-Port pull-up resistor can be built in the R-Port by shorting the corresponding RPU0 ~ RPU3 with the side of H. If installation of built-in pull-up resistor is not desired, short them with the side of L.

Setting pin	R0	R1	R2	R3
Short post	RPU0	RPU1	RPU2	RPU3
Built-in pull-up resistor installation	H	H	H	H
No built-in pull-up resistor installation	L	L	L	L

3. Setting of output condition of D0~D7 in STOP
   Shorting the DSC0~DSC7 with the side of H can set the output condition of corresponding D-output in STOP at ″L″ forcibly.
   To set the condition of usual output (the condition before STOP started is maintained), short them with the side of L.

Setting pin	D0	D1	D2	D3	D4	D5	D6	D7
Short post	DSC0	DSC1	DSC2	DSC3	DSC4	DSC5	DSC6	DSC7
Forced setting at ″L″ in STOP release	H	H	H	H	H	H	H	H
Usual output in STOP released	L	L	L	L	L	L	L	L

4. Setting of watch dog timer release with REMOUT output
   The watch dog timer can be reset with REMOUT output signals by shorting the WDTM with the side of L.
   If the WDT resetting with REMOUT output signals is not desired, short it with the side of H.

Short post	WDTM
Reset timer	L
Do not reset timer	H

[S2] External STOP setting
   STOP can be set from the outside by shorting the S2 toward the side of H.
   Usually, short it toward the side of L.

[S3] Power supply connection
   This selection should be strapped to V_{DD}.

[S4] EPROM 2764/128, and 27256 can be installed by switching over the S4. For EPROM, however, right-justify ROM chip pin 1 from socket pin 3.

Short post	S4
2764/128	H
27256	L

[S5, S6] Clock input selection

Self-induced oscillation with the external clock input and oscillator can be set by switching over the S5 & S6

Short post	S5 & S6
External clock input	U
Internal self-induced oscillation	X

For internal clock input, install an oscillator on the PCB. Since the oscillation circuit constant varies depending on the oscillator, adjust the constant by referring to the oscillator manufacture`s recommendable values.

[S7, S8, JP] Clock input selection

MHz and KHz oscillation can be selected by switching over the S7, S8 and JP.

Short post	S7 & S8 & JP
MHz oscillation	M
KHz oscillation	K

Caution on Operation

- It is required to install a 24DIP IC socket in the application system. The connection cable is connected to the socket.
- There is a possibility that the ceramic oscillator on the application system cannot oscillate properly due to the influence of connection cable wiring capacitor or other reasons. In such a case, install the oscillator on the evaluation board.
- Since the GMS 30000 EVA is designed to evaluate the program operations, there is a case where the AC and DC characteristics differ from those of the mass-produced chips

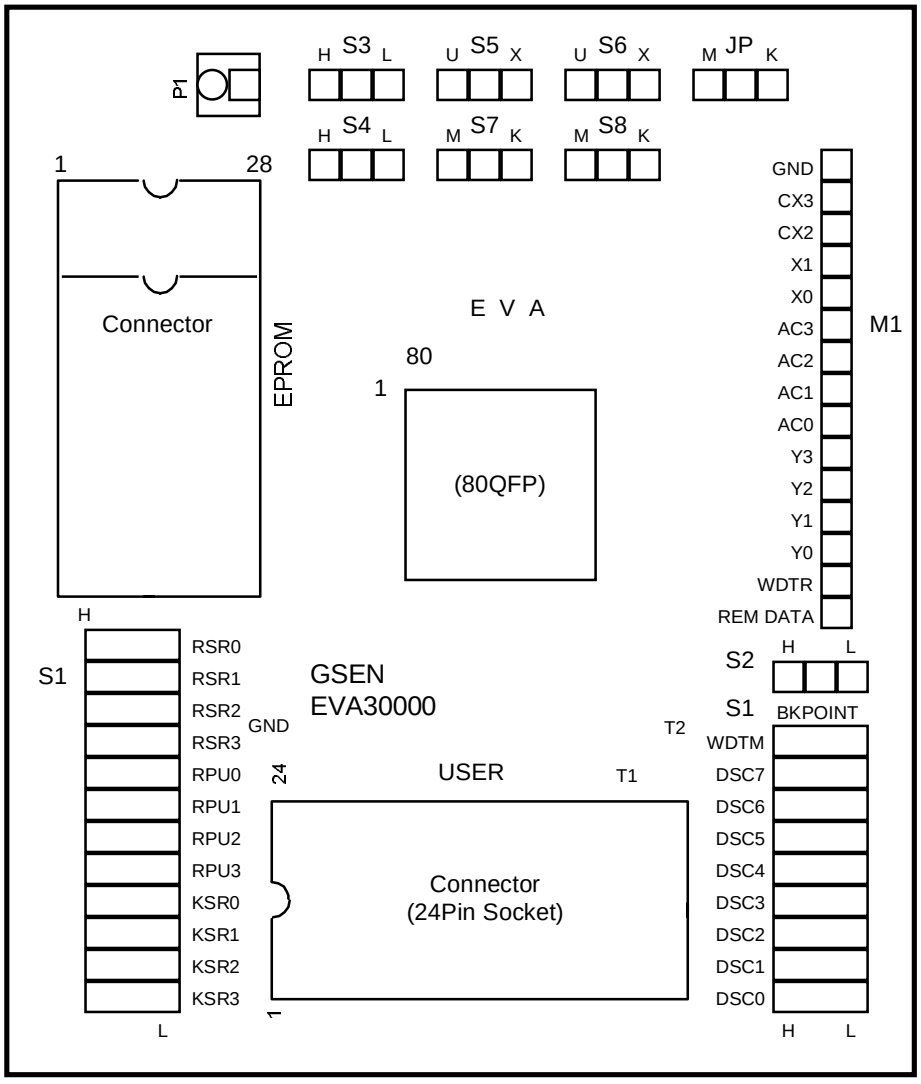


Fig 4-1 Layout Diagram

INTRODUCTION	1
ARCHITECTURE	2
INSTRUCTION	3
EVALUATION BOARD	4
SOFTWARE	5
APPENDIX	6

CHAPTER 5. Software

Configuration of Assembler

Execute File	Description
GA80.EXE	Assembler
GMSLST.EXE	Create assembler list file
GMSHEX.EXE	Create HEX.file
GMSCRF.EXE	Create cross reference file
GMSTST.EXE	Create instruction check file
GMSROM.EXE	Create ROM dump file
GS.BAT	Batch processing of the above
GMS30K.LIB	Instruction library file

Boothing up Assembler

Creating your own source file with the extension name of SRC and execute batch file (GS.BAT). This batch file converts the source code written in mnemonic into machine language and generate a kind of useful file.

C> GS Source file (.SRC)

Input File	Output File	Content
EX.SRC	EX.LST	List file
	EX.RHX	Hexa file (for EPROM, simulator)
	EX.CRF	Cross reference file
	EX.TST	Instruction check file
	EX.DMP	ROM dump file (for masking data)
	EX.SYM	Symbol file

* HEX and PRN file is intermediate file

Configuration of Simulator

1. Overview

The simulator is a program for GMS300 Series 4-bit one-chip microcomputer. The environment is organized based upon Hexa file of *.RHX and Cross Reference file of *.CRF generated by assembling the source program coded by programmer.

Execution Environment

System : IBM-PC/AT or higher (MS-DOS or PC-DOS)
Video : Hercules, EGA or VGA color

Organizing files

```
GSSIM.EXE      :  Simulator execution file
GMS30K.GSP     :  Store the simulator environment.  It is
                  generated automatically when executing
                  the program initially (Selected CPU.
                  Store the file names previously loaded.).
GMS30K.HLP     :  Help file of simulator commands.
GMS30K.LOG     :  Record the working history of users.  Generated
                  by LOG ON and LOG OFF commands.
*.BAT          :  List a set of simulator command.  Generated by
                  user.
PORTIN.DAT     :  Provide the port input-value when executing
                  the simulator.  Generated by user.
```

Supporting CPU

GMS30004, GMS30012, GMS30112, GMS30120, GMS30140

2. Characteristics of Simulator

- User-friendly pop-up window menu. Select the necessary command and display the screen in windows format so that users can know the execution results.

- Display always the register window in the right side so that programmer can check easily the change of data memory value as program proceeds.

- Maintain the previous simulator environment if user does not make the extra changes when re-executing after logging out completely from the simulator previously executed by loading the source program. In other words, the previously-executed file is automatically loaded when the simulator is executed (GMS30K.GSP file).

- When trace command ([F8] or > T command) is executed, the changed values are noticed easily by displaying the highlighted changed values in register and memory windows, if the contents of each register or data memory is changed as command line is processed.

- When trace command ([F8] or > T command) is executed, the current execution line is highlighted.

- Out of the simulator commands, load or save commands is executable in pop-up windows. In-line command is executable as prompt command in the command window.

3. Screen Organization of Simulator

Screen is basically organized with four windows; Memory, Source, Command and Register. Source and Command windows can be enlarged up to the full screen size (CTRL-[F10]). Movement between windows is made by [F6].

3.1 Memory Window

Data memory contents of the currently selected micom is displayed. 32 nibble data values of 00~1F(h) addresses are displayed.

3.2 Source Window

The contents of *.RHX file called by load command is displayed in the state of being disassembled. Addresses are displayed randomly in the state of polynomial together with instruction code and mnemonic. If *.CRF file is called, label is displayed at the corresponding position. Display position of source program is adjustable with Up/Down arrow keys and Page Up/Down keys.

3.3 Command Window

All kinds of commands provided by simulator is executed by In-line command, and the execution results of the commands are displayed. Command window size is adjustable with [CTRL]-[F10].

3.4 Register Window

Display each register value inside micom, I/O port value and machine cycle altogether. When trace command is executed by function key [F8], the register value after the previous command before the current program counter is executed is displayed. All kinds of register and I/O ports displayed in register window are as follows.

```
PC     : Program counter       2digit   6bit     [Hexa]
ACC    : Accumulator           1digit   4bit     [Hexa]
PA     : Page address          1digit   4bit     [Hexa]
PB     : Page buffer           1digit   4bit     [Hexa]
X      : X register            1digit   1 or 2   [Hexa]
Y      : Y register            1digit   4bit     [Hexa]
ST     : Status register       1digit   1bit     [Binary]
PMR    : Pulse mode register   1digit            [Hexa]
WDT    : Watch dog timer       1digit   14bit    {Hexa]
SP     : Stack pointer level   1digit
SR0    : Stack level 0         4digit
SR1    : Stack level 1         4digit
SR2    : Stack level 2         4digit
OUT    : Remocon out output                     [Binary]
K      : K port input register          4bit     [Binary]
Rin    : R port input register                   [Binary]
Rout   : R port output register         [Binary]
D      : output port                6or8 10bit  [Binary]
Machine Cycle :
    The number of command execution is displayed in decimal.
```

4. Commands in Each Menu

^stands for [CTRL] key, while @ stands for [ALT] key.

4.1 File Menu

Use the function key behind each command as a hot key or execute each command through selecting [ALT]-[F] key and pressing the highlighted character.

Load RHX F2 : Load the file named *.RHX, analyze the selected Hexa file and disassemble it. Display the program address, assemble code and mnemonic. The order of displayed program addresses follows the POLYNOMIAL form. Even when the extention is not input in case of selection, .RHX extension is presumed to include.

Load CRF @F2 : If Cross reference file of loaded file loads the *.CRF file, labels and variables assigned by programmer are displayed at accurate position of Source window so that programmer can read the program easily. Even when the extension is not input in case of file selection, .CRF extension is presumed to include.

Write RHX F3 : When any modifications are made to source program or program memory after the simulator is loaded once, the modifications are stored in the same or new filename as loaded. It has the same command and function as > WP [filename] of In-line command.

Log ON/OFF F4 : After the simulator is executed, all the input and results are stored in the filename GMS30K.LOG Once function key [F4] is pressed, log-in starts, and if the key is pressed one more time, log-in file is closed in a toggling way. The ON/OFF state of log-in is displayed in the upper-right corner. It has the same function as > LOGON and > LOGOFF of In-line commands.

Os shell @S : When users want to work temporarily under DOS environment, this command is used. When users want to back to Windows environment, input > EXIT.

Exit @X : Used when getting completely out of the simulator environment.

4.2 Window Menu

Use the function key behind each command as a hot key or execute each command through selecting [ALT]-[W] key and pressing the highlighted character. Function key [F6] provides the return function to each window.

Command Box: Position the cursor in command window to make it possible to use In-line command provided by the simulator.

Source Box : Position the cursor in source window to make it possible for programmer to see the disassembled source program. It is possible to use Up/Down arrow keys and Page Up/Down keys.

Zoom ^F10 : Position the cursor in command or source window, and then select Zoom or press [CTRL]-[F10] key to enlarge the window to the full screen size.

4.3 Run Menu

Use the function key behind each command as a hot key or execute each command through selecting [ALT]-[R] key and pressing the highlighted character.

MCU Reset ^F9 : Initialize the execution environment of the simulator. In other words, initialize the register value to 0 or 1, and machine cycle value is changed to 0. It has the same command and function as > CR command of In-line command.

Go F5 : Program executes from the current value of program counter. Press [ESC], [Enter] or [Space] key to stop execution, and display the current register value.

Animate @5 : Program executes from the current value of program counter. The value of data memory or registers are highlighted in the corresponding window. Press [ESC], [Enter] or [Space] key to stop execution. Because of speed difference among system, the speed is adjustable from 0 to 40.
(0 : fastest, 40 : slowest)

Trace F8 : Program executes line by line from the current value of program counter. The changing values of registers and memory are highlighted in register window and memory window respectively.

Execute Batch : When the batch filename consisted of a set of commands made by user using editor is input, each command executes automatically as In-line command is input. It has the same function as > BAT command of In-line command.

4.4 Option Menu

To execute each command, select [ALT]-[O] key and press the highlighted character in each command line. Or select the menu and press the [Enter] key.

MCU Select : Select according to the kind of micom. Able to select on of GMS30004, 30012, 30112, 30120 or 30140 among GMS300 series. Once the command is executed, the window indicating the characteristics of each micom is open. Press Left, Right, Up Down key to select the micom to work.

Setup : Set the execution mode of selected micom. It has the same function as > SET command of In-line command. Once the function is executed, a small <Setup> window is open. Position the cursor in either of I/O Input and Output mode, Symbol, Execution Mode of Watch Dog Timer with the item to change. Assign the corresponding execution mode with Left, Right arrow key.

Execution Mode to be Assigned in <Setup>

I/O Input [pi] = [0] Port Input from I/O register
[1] Port input from keyboard
[2] Port input from file (PORTIN.DAT)

I/O Output = [0] No display
[1] Display
[2] Display & Break

Symbol = [0] Search
[1] Unsearch

Watch Dog Timer = [0] OFF
[1] ON (No option)
[2] ON (Option)

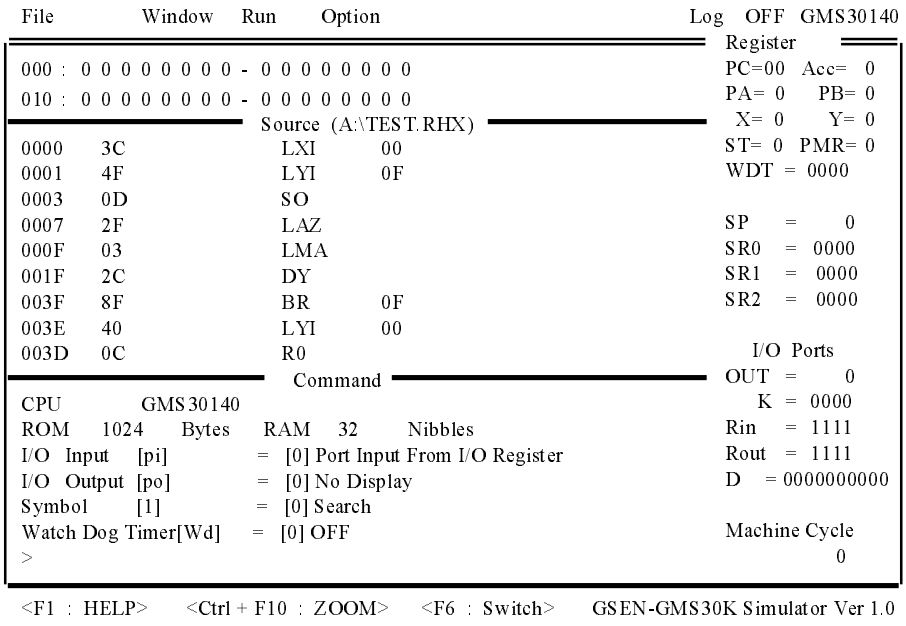
5. Simulator Execution

5.1 File Load

Use one of three ways to load the file to run from the simulator. First execute the GSSIM.EXE file from DOS and name it as a parameter.

>a:\GSSIM TEST.RHX (Here the extention needs not be input.)

The following screen will be displayed.



Second, execute the GSSIM.EXE file and then the Load RHX (Hot key is [F2]). The following small window is open at the center of screen waiting for user to input the Hexa filename to work.

When no file is selected, Unnamed.rhx filename is displayed. When the filename to work is input immediately, the file from the current directory is called. If the specific directory is assigned, the file from the assigned directory is called.

Third, call the file to work through using the >LP [filename] from command window by in-line command.

Here for example, load the TEST.RHX file

TEST.RHX file is called and the contents of HEXA file is analyzed from the simulator. The source contents in the state of being disassembled is displayed from Source window. Incase of filename input, although RHX is not input, .RHX extension presumed to include by default.

Also when there is Cross Reference File of working file, press Load CRF (Hot key [ALT]-[F2]). The following small window is open at the center of screen waiting for user to input *.CRF filename.

The filename called by Load RHX command is displayed by default as .CRF filename. When .CRF file is called, the label of source program created by programmer is displayed at the label position of Source window for easy reading of program by user.

5.2 File Store

When the specific part of source program is changed under the simulator environment by calling the working file, the corresponding Hexa file needs to be stored. Use one of the following two methods.

First, when executing the pop-up command Save RHX (Hot key is [F3]), the following small window is open at the center of screen waiting for user to input the Hexa filename. The filename called by file load command by default is displayed. When the filename is not changed, hit just the [Enter] key. When user wants to change the filename to store, input the filename to change.

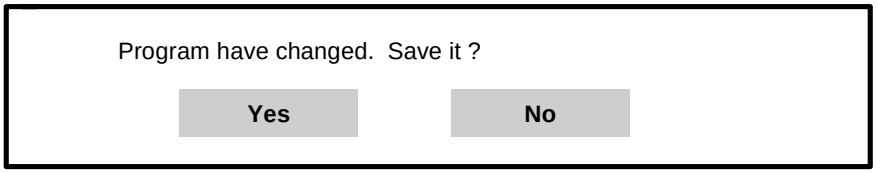
Second, store the processed Hexa file using > WP [filename] from command window with In-line command.

Here when the same filename to store already exists in the disk, ˝File Already Exist˝ message comes up asking user by [YES/NO] if user overwrite or not.

5.3 Closing the Simulator

Using the pop-up command Exit (Hot key [ALT]-[X]) or In-line command > Q, exit from the simulator environment.

When execute the command, the following message comes up for the check-up asking the user`s intention to store, if user does not store the changed file after changing the loaded file to work. Use Tab key or left, right direction key to select YES/NO.



Also for recording the work contents, even when exiting the simulator without Log OFF, Log OFF is done automatically and GMS30K.LOG file is stored.

6. Simulator Commands

6.1 Command Syntax

1) A (Assemble)
To assemble what is commanded for every line from the specified <address> and write in the memory.

2) BAT (Batch)
To execute what is commanded in the command file in a batch.
When there is a format error, command error is issued and execution is stopped at the error point.

3) BP (Break Point set), BL (Break point List)
To set break point.
To display the set break Point.

4) BS (Break point set step)
To set break point with No. of steps.

5) BC (Break point Clear)
To clear the specified No. break point.

6) CR (CPU Reset)
To reset the simulator to the initial state.

7) DPP (Dump Program memory)
To display the content of the memory in the area of the No. of pages specified with <In> from the specified <address> in hexadecimal. The address here is polynomial.

8) DPS (Dump Program memory)
To display the content of the memory in the area of the No. of pages specified with <In> from the specified <address> in hexadecimal.

9) DD (Dump Data memory)
To display all the data in Data Memory in hexadecimal.

10) EPP (Exchange Program memory)
To display and modify the specified data in the program memory.
Address here is polynomial.

11) EPS (Exchange Program memory)
To display and modify the specified data in the program memory.

12) ED (Exchange Data memory)
To display or modify data in the specified data memory.

13) FPP (Fill Program memory)
To fill the area of the program memory specified with <In> from the specified <address> with the specified byte data. The address here is polynomial.

14) FPS (Fill Program memory)
To fill the area of the program memory specified with <In> from the specified <address> with the specified byte data.

15) FD (Fill Data memory)
To fill the area of the data memory specified with <In> from the specified <address> with the specified nibble data.

16) G (Go)
To execute the program in the specified program memory.

17) H (Hex calculate)
To add or subtract in hexadecimal.

18) LOGON (LOGIN)
To log the commands executed after this command.

19) LOGOFF (LOGOUT)
To end logging.

20) LP (Load Program from MS-DOS* file)
To load `files` on MS-DOS* to the memory.

21) MPP (Move Program memory)
To transfer data in the memory area to another area.
The address here is polynomial.

22) MPS (Move Program memory)
To transfer data in the memory area to another area.

23) P (Port set)
To set the specified data at the specified I/O register.

24) Q (Quit)
To return to MS-DOS*.

25) R (Register dump or change)
To display or modify the register data.

26) SET (setup)
To set the operation Mode for the simulator.

27) SL (Symbol file Load)
To load symbol tables from the specified symbol file.

28) ST (Status)
To display the simulator status.

29) T (Trace)
To execute the program in the specified program memory address a single step.

30) TMT(Time)
To obtain time from the No. of machine cycles and clock frequency.

31) TMC (Time)
To obtain No. of machine cycle from the time and clock frequency.

32) U (Unassemble)
To unassemble data in the area specified with <In> from the specified <address> and display in mnemonic.

33) Wp (Write Program to MS-DOS* file)
To read-out data in all the ROM area and write the Intel hexa data in the files specified with <file name>.

34) ? (Help)
To display the list of commands of this simulator

6.3 Description of Commands

The symbols used in this chapter are defined as in below.

1) XXXX Indicates input from the keyboard.
2) ↓ Indicates Return key.
3) _ Indicates insertion of space characters.
4) In Indicates range.
5) [ ] Indicates it is omittable.
6) No. used on this system are hexadecimal. However, the machine cycles are decimal.

(Common items on this simulator)

1) This simulator accepts up to 132 characters per each line.
   Before pressing `↓` key, data can be modified in the following procedure.
   <BS> Key deletes one character and the cursor sets back by one character.

2) Commands can be input both in block and small letters and both are treated as the same.

3) Commands can be terminated by inputting `.`.

4) Command history (recall)
   This simulator has 16 command buffers and each time `Control A` is pressed, command immediately before the current one is displayed.

5) When `!XX` is executed to `*`, MS-DOS* commands can be executed temporarily.

6) Regarding the omission indicated with [ ], when one [<address>] or [In] is omitted, the area may not be recognized. In this case, be careful as the current operation cannot be ensured.

7) When `.XXXX` is specified, it is treated as a symbol. Therefore, before using this specification, specification of Symbol file should have been made.

8) When `ESC` key is pressed, command execution is stopped and the system moves to `*` mode.

9) When In is LXX< XX indicates No. of words and when there is no L, XX indicates an address.

10) For command separator (indicated with_in this Manual), `_` or `,` can be used.

11) When `Control C` is pressed, the system goes back to `MS-DOS*`.

12) In case of both sequential and polynomial address, the first two digits of an <address> indicate No. of page and the latter two digits indicate the address.

Hereunder is the explanation on each command used on this system.

A (Assemble)

[Function]
To assemble commands for each line from the specified <address> and write them in the memory.

[Format]
> A_ [<address>] ↓

[Explanation]
With this, the system assembles commands for each line from the specified <address> and writes them in the memory.
When <address> is omitted, data is written from the current `PC` address.
Assemble can be finished by keying in `.`.
When `_` is keyed in, the system goes back the address just before the current one.

[e.g.]

```
>A 200 ↓
0200    40   LYI     0      SO ↓
0201    21   LAM            LMA ↓
0203    77   ALEI    14     - ↓
0201    03   LMA            - ↓
0200    0D   SO             . ↓
>
```

BAT (Batch)

[Function]
To execute commands in the command file in a batch. When there is a format error, execution is stopped there and command error is issued.

[Format]
BAT_<File Name> ↓

[Explanation]
With this command, the system executes commands in the command file in a batch. When there is a format error, execution is stopped there and command error is issued. In order to execute this command, it is required to create the command file on the editor in advance.

[e.g]

```
>BAT test.bat ↓
>R  PA  0
>R  PB  0BATCH END
>
```

BP (Break Point set), BL (Break point List)

[Function]

BP	To set the break point.
BL	To display the set break point
BPS	To set step break point.

[Format]

BP[n]_adr[_m]↓

n :	Set break No. 0~9.
adr :	Set address for setting break
m :	Valid only when n=0, setting No. of times to pass the break point. The m range is 1≤m≤255. m value should be set in hexadecimal.

BPS_st↓

st :	Set No. of steps. st range is 1≤st≤2147483647. st value should be set in decimal.

BPI↓
BPO↓
BL↓

[Explanation]

This command is used to the break point.
Break on this simulator is to stop after executing the command on the specified address.
When `BPS st` is specified, the system stops when the value on the machine cycle register becomes equal to st.
When `BPI` is specified, the system stops before command execution every time command input is made.
Unassemble displayed in this case is the address with input command.
When `BPO` is specified, the system displays the value on that occasion on CRT and stops every time output command is made.
When `BP` only or `BL` is specified, the state of the currently set break point is displayed.
It is possible to used symbols for specifying adr.
When `n` is omitted in break setting command, No. shall be allocated from 1 to all empty No.

[e.g.]

1) Example to occur break after passing page 1 address 0 three times.

```
>BP0   100   3↓
```

2) Example to set a break point at page 5 address 0.

```
>BP    500   ↓
>BL ↓
 0  =  0100   ( 3)
 1  =  0500
```

3) Example to occur break on the main label (in case the main label is at page 5 address 0.)

```
>BP    .main  ↓
>BL ↓
 0  =  0100   ( 3)
 1  =  0500  .MAIN
>
```

4) Example to occur break by No. of steps (50 steps).

```
>BPS   50↓
>BL ↓
 0  =  0100   ( 3)
 1  =  0500  .MAIN
 S    50
>
```

5) Example to occur break with input command.

```
>BPI ↓
>G ↓
RUN
********  MC Step Break ********
0004      00                NOP
PC=00  PA=09  PB=00  A=0  X=0  Y=0  ST=1  PMR=0  WD=0000  MC=50
>R   R   00   ↓
>G   ↓
```

6) Example to occur break with output command.

```
>BP0   ↓
>G ↓
RUN
Rout=1111    D=11111111    OUT=0      MC=125
********  Break AT Port Out !!  ********
002F   0D      SO
PC=00  PA=1E  PB=00  A=0  X=0  Y=F  ST=1  PMR=0  WD=0000  MC=125
>G   ↓
```

BC (Break Point Clear)

[Function]
To clear the break point at the specified No.

[Format]
BC_n↓

[Explanation]
With this, the system clear the break point at the specified No.
When n is omitted, the state of the currently set break shall be displayed.
When n is `*`, all the break points shall be cleared.

[e.g]

```
>BL ↓
0  =  0100    (  3)
1  =  0200
2  =  0500
S  50
I
O
>BC  1  ↓
>BC  S  ↓
>BC  I  ↓
>BC  O  ↓
>BC ↓
0  =  0100    (  3)
2  =  0500
>BC *  ↓
>BL ↓
>
```

CR (CPU Reset)

[Function]
To set simulator to the initial state.

[Format]
CR↓

[Explanation]
When this command is executed, the simulator is set back to the initial state. In this case, the registers are also initialized according to each CPU. However, MC is cleared irregardless of the CPU status.
Initial State of each register.

PA, PC, PB, SP and D PORT become 0.
All the ports of R PORT become 1.
The other registers are undefined.

[e.g]

```
>CR ↓
CPU         GMS30140
ROM   1024    Byte         RAM 32 Nibble
I/O  Input  [pi]           = [0]  Port input I/O register
I/O  Output [po]           = [0]  No Display
SYMBOL           [1]       = [0]  Search
Watch Dog Timer [wd]       = [0]  off

PC=00  PA=00  PB=00  A=0  X=0  Y=0  ST=0  PMR=0  WD=0000  MC=0
                    SP=0  SR1=0000  SR2=0000  SR3=0000
I/O  Reg.  Rin=1111  Rout=1111  D=00000000  OUT=0  K=0000(0h)
>
```

DPP (Dump Program memory)

[Function]
To display data in the program memory area specified with <In> from the specified <address> in hexadecimal.

[Format]
DPP_[<address>_<In>]↓

[Explanation]
When this the system displays data in the program memory area specified with <In> from the specified <address> in hexadecimal. When [ ] is omitted, 64 bytes of data from the succeeding address shall be displayed. The address used with this command is polynomial.

[e.g]

```
>DPP   0   20  ↓
0000  :  00 01 02 03 04 05 06 07 - 08 09 0A 0B 0C 0D 0E 0F
0027  :  10 11 12 13 14 15 16 17 - 18 19 1A 1B 1C 1D 1E 1F
001C  :  20 21 22 23 24 25 26 27 - 28 29 2A 2B 2C 2D 2E 2F
0022  :  31 32 33 33 34 35 36 37 - 38 39 3A 3B 3C 3E 3E 3F
>
```

DPS (Dump Program memory)

[Function]
To display data in the program memory area specified with <In> from the specified <address> in hexadecimal.

[Format]
DPS_[<address>_<In>]↓

[Explanation]
When this the system displays data in the program memory area specified with <In> from the specified <address> in hexadecimal. When [ ] is omitted, 64 bytes of data from the succeeding address shall be displayed. The address used with this command is sequential.

[e.g]

```
>DPS_0_3F  ↓
0000  :  00 01 03 07 0F 1F 3F E3 - 3D 3B 37 2F 1E 3C 39 33
0010  :  27 0E 1D 3A 35 2B 16 2C - 18 30 21 02 05 0B 17 2E
0020  :  1C 38 31 23 06 0D 1B 36 - 2D 1A 34 29 12 24 08 11
0030  :  22 04 09 13 26 0C 19 32 - 25 0A 15 2A 14 28 10 20
>
```

DD (Dump Data memory)

[Function]
To display data in the data memory in hexadecimal.

[Format]
DD↓

[Explanation]
With this, the system displays all the data in the data memory in hexadecimal.

[e.g]

```
>DD ↓
000  :  6  C  6  0  0  0  0  0  -  0  0  0  0  0  0  0  0
010  :  1  2  3  4  0  0  0  0  -  3  2  1  0  0  1  0  1
>
```

ED (Exchange Data memory)

[Function]
To display and modify the specified data memory.

[Format]
ED_[<address>↓

[Explanation]
When this the system displays and modifies the specified data memory. This Mode is continuous and processing shall be continued until *.* is pressed. When `_` is pressed, the system goes back to the address immediately before.

[e.g]

```
>ED  0 ↓
00  :  7  >  3↓
01  :  6  >  6↓
02  :  7  >  .↓
>
```

EPP (Exchange Program memory)

[Function]
To display and modify the specified program memory.

[Format]
EPP_<Address> ↓

[Explanation]
With this, the system displays and modifies the specified program memory. This mode is continuous and each time `↓` is pressed, the succeeding address shall be displayed, setting operation shall be performed continuously until `.` is pressed. When `_` is pressed, the system goes back to the address immediately before. The address used with this command is polynomial. It is possible to use symbols in <address>.

[e.g]

```
>EPP_100 ↓
100  :  0D>37 ↓
101  :  77>AF ↓
103  :  42> . ↓
>
```

EPS (Exchange Program memory)

[Function]
To display and modify the specified data memory.

[Format]
EPS_<Address>↓

[Explanation]
With this, the system displays and modifies the specified program memory. This Mode is continuous and each time `↓` is pressed, the succeeding address shall be displayed and setting operation shall be performed continuously until `.` is pressed. When `_` is pressed, the system goes back to the address immediately before. The address used with this command is sequential.
It is possible to use symbols in <address>.

[e.g]

```
>EPS_100 ↓
100  :  48>6↓
101  :  04>45↓
102  :  53>.↓
>
```

FPP (Fill Program memory)

[Function]
To fill the program memory area specified with <In> from the specified <address> with 1 byte data.

[Format]
FPP_<Address>_<In>_<Data> ↓

[Explanation]
With this, the system fills the program memory area specified with <In> from the specified <address> with 1 byte data. It is possible to use symbols in <address>.

[e.g]

```
>FPP  100  L10  55 ↓
>DPP  100 ↓
0100 :  55 55 55 55 55 55 55 55 - 55 55 55 55 55 55 55 55
0127 :  10 11 12 13 14 15 16 17 - 18 19 1A 1B 1C 1D 1E 1F
011C :  20 21 23 24 25 26 27 29 - 29 2A 2B 2C 2D 2E 2F 30
0122 :  31 32 33 34 35 36 37 38 - 39 3A 3B 3C 3D 3E 3F 40
>
```

FPS (Fill Program memory)

[Function]
To fill the program memory area specified with <In> from the specified <address> with 1 byte data.

[Format]
FPS_<Address>_<In>_<Data>↓

[Explanation]
With this, the system fills the program memory area specified with <In> from the specified <address> with 1 byte data.
It is possible to use symbols in <address>.

[e.g]

```
>FPS  100  L10  55 ↓
>DPS  100 ↓
0100 : 55 55 55 55 55 55 55 55 - 55 55 55 55 55 55 55 55
0110 : 01 00 11 01 01 20 00 00 - 01 00 00 0C 01 00 55 55
0120 : 01 00 11 01 01 20 00 00 - 01 00 00 0C 01 00 00 55
0130 : 01 00 11 35 01 20 00 55 - 01 55 00 55 55 55 55 55
>
```

FD (Fill Data memory)

[Function]

To fill the program memory area specified with <In> from the specified <address> with one specified nibble data.

[Format]

FD_<Address>_<In>_<Data> ↓

[Explanation]

With this, the system fills the data memory area specified with <In> from the specified <address> with one specified nibble data. It is possible to use symbols in <address>.

[e.g]

```
>FD  0  L3  4 ↓
>DD  ↓
000 :  4  4  4  0  0  0  0  0  -  0  0  0  0  0  0  0  0
010 :  0  0  0  0  0  0  0  0  -  0  0  0  0  0  0  0  0
>
```

G (Go)

[Function]
To execute the program in the specified program memory

[Format]
G_[<Address 1>][_<Address 2>]...[_<Address 8>] ↓

[Explanation]
With this, the system executes the program address specified with =, [=<address 1>] is omittable. When omitted, the command is executed from the present `PC` address. The system also sets a break point in the specified address after [_<address2>] . When `g` command is executed, simulation is started after outputting `RUN` message. When any key is pressed in this state, simulation is stopped.
It is possible to use symbols in the <address>.

[e.g]

1) **>G ↓**

```
RUN
******** User Break Point !! ********
010E  20           LMAIY
PC=01  PA=1D  PB=01  A=7  X=1  Y=8  ST=1  PMR=0  WD=0000  MC=297
>
```

2) **>G=.START 37 3↓**

```
******** Go Break Point !! ********
0003  BF             BR      3F
PC=00  PA=07  PB=06  A=F  X=3  Y=5  ST=1  PMR=0  WD=0000  MC=808
>
```

H (Hex calculate)

[Function]
To add and subtract hexadecimal No.

[Format]
H_XXXX_XXXX ↓

[Explanation]
Hexadecimal Nos. are added or subtracted.

[e.g]
```
>H_e6ab_b7fc ↓
 9ea7   2eaf
>
```

LOGON (LOGIN)

[Function]
To log the commands executed after this command.

[Format]
LOGON↓

[Explanation]
After executing this command, all the information displayed on CRT shall be written consecutively until LOGOFF command is given. The file created in this event is ˝LOG.DAT`.

[e.g]
```
>LOGON ↓
>
```

LOGOFF (LOGOUT)

[Function]
To end logging

[Format]
LOGOFF↓

[Explanation]
Logging is finished when this command is executed.

[e.g]
```
>LOGOFF↓
>
```

LP (Load Program from MS-DOS* file)

[Function]
To read `file` on MS-DOS* and write it to the program memory of the simulator

[Format]
LP_<file name>↓

[Explanation]
The system reads the file specified with <file name. RHX> and writes the data to the address specified with <address>.
The file format is the Intel hexa format.

[e.g]
```
>LP TEST.  RHX ↓
. . . .
```

MPP (Move Program memory)

[Function]
To transfer memory area data to other area.

[Format]
MPP_<address_s>_<In>_<address_d>↓

[Explanation]
With this command, the system transfers data upto the No. of words specified with <In> from the address specified with <address_s> to the area specified with <address_d>.
The address used with this command is polynomial.

[e.g]
```
>MPP  100  200  300↓
>
```

MPS (Move Program memory)

[Function]
To transfer memory area data to other area.

[Format]
MPS_<address_s>_<In>_<address_d>↓

[Explanation]
With this command, the system transfers data upto the No. of words specified with <In> from the address specified with <address_s> to the area specified with <address_d>.
The address used with this command is sequential.

[e.g]
```
>MPS  100  200  300↓
>
```

P (Port Set)

[Function]
To display and modify the specified I/O set registers.

[Format]
P_aa_bb_c↓ (When setting R, D port)
- aa : I/O set register name
- bb : Specify in or out
- c : Set value (0 or 1)

P_aa_cc ↓ / P_aa_c↓ (when setting K)
- aa : I/O set register name
- bb : set value (one digit of a hexadecimal No.)
- c : Set value (0 or 1)

[Explanation]
The system sets the value in the specified I/O set register.

[e.g]
1) **Example of setting K**

```
> P   K   8 ↓
> P↓
I/O Regs.  Rin=0000  Rout=1111  D=00000000  OUT=0  K=1000(8h)
>
```

2) Example of K0-K3 setting

```
> P   OUT  1↓
> P   K1   1↓
> P ↓
I/O Regs.  Rin=0000  Rout=1111  D=00000000  OUT=1  K=0010(2h)
>
```

(I/O setting registers used on this simulator)
- Dout (D port output register 6 or 8 or 10bit)
- K (K input register 4bit)
- Rout (R port output register 4bit)
- Rin (R port input register 4bit)
- OUT (OUT port output register 10bit(

Q (Quit)

[Function]
To return to PC-DOS

[Format]
Q↓

[Explanation]
When this command is executed on the system with prompt (*display) waiting for a command, the system moves from `simulator` Mode to `MS-DOS*` Mode.

[e.g]
1)**Q ↓**
A>

R (Register dump or change)

[Function]
To display and modify the register data.

[Format]
R↓
R_x↓
R_a=x↓

[Explanation]
With this, the register data is displayed and modified.
When `R↓` only is executed, all the registers are displayed.

[e.g]
```
>R ↓
PC=00  PA=3E  PB=00  A=0  X=0  Y=1  ST=1  PMR=0  WD=0000  MC=10392
                     SP=0  SR0=0000  SR1=0000  SR2=0000

>R  PC=23 ↓
>R  PC ↓
PC=23
>R  MC =0 ↓
>R  MC ↓
MC=0
>
```

(Registers used on this simulator)

Simulator setting registers

- MC (Machine cycle register 32bit)

GMS300 series

- PC (Program counter 6bit)
- PA (Page address register 4bit)
- PB (Page buffer register 4bit)
- A (Acc register 4bit)
- X (X register 2bit)
- Y (Y register 4bit)
- SP (Stack pointer register 2bit)
- SR (Stack register 10bit) ×3
- ST (Status register 1bit)
- PMR (Pulse mode register 4bit)
- WD (Watch dog timer register 14bit)

SET (SETUP)

[Function]
To setup the operating mode for the simulator.

[Format]
SET_c_x↓

[Explanation]
This is used to setup the simulator.
The following commands can be executed with this command.

• When setting symbols for Assembler and Unassembler.
SET L x ↓
Here, the range of X is defined as in the below.
0 : Symbol shall be used. (Default)
1 : Symbol shall not be used.

• SET PO x ↓
0 : When I/O WRITE command is executed while executing G command, no display shall be mode. (Default)
1 : When I/O WRITE command is executed while executing G command, the values in the event shall be displayed on CRT screen.
2 : When I/O WRITE command is executed while executing G command, the values in the event shall be displayed on CRT screen and the operating shall be stopped.

• SET PI x ↓
0 : In case I/O READ command is to be executed while simulating, the system reads the value set on the port input register. The port input register setting is to be performed with `R` command.
1 : In case I/O READ command is to be executed while simulating, the system stops before the execution.
In this case, set the value on the port input register.
2 : Value is set on the port input register from I/O file (PORT.DAT).

The file is opened when this command is executed and the file shall not be reread. If there is no I/O file, error shall be issued.

Timing for setting port input register is :

a) When machine cycle of the file <current machine cycle after executing this command, values shall be set on the port input register until the machine cycle of the file> current machine cycle while executing the first G or T command.
b) When machine cycle of the file = current machine cycle, setting is made immediately before the next command is executed.
In case there is a format error in I/O file while simulating, error message shall be output and the operation is stopped.
In order to execute G or T command again, this mode has to be canceled first.
(Execute SET PI 0 or SET PI 1.)
Default value here is 0.
All the I/O READ commands are read from the port input register.
PORT.DAT format
Machine cycle I Port Name I Data ↓
(I indicates space or tab)

• SET WD x ↓

0 : Watch dog timer shall not be used.

1 : Watch dog timer shall be used. (No option)
Resetting watch dog timer

1) After executing WDTR command
2) After executing STOP command
3) After executing CR command
4) While converting to 0 with R command
5) When the set value is reached.

2 : Watch dog timer shall be used. (With option)
For resetting watch dog timer.

1) When SO command is executed to REMOUT output is added to the above 1) ~ 5).

Defalut value here is 0.

[e.g]

1) Example of setting port input I/O file and measure in case of file format error.

```
>SET  PO  1 ↓
>SET  PI  2 ↓
>SET ↓

CPU    GMS30140
ROM   1024    Byte        RAM    32   Nibble
I/O  Input  [pi]     = [2]  Port Input file  (PORT.DAT)
I/O  Output [po]     = [1]  Display
Symbol      [1]      = [0]  Search
Watch Dog timer [wd] = [0]  off
>G ↓
RUN
********  `PORTIN.DAT`  File format error ********
0001                07     DA
PC=0  PA=01  PB=00  A=0  X=0  Y=0  ST=1  PMR=0  WD=0000  MC=235
```

In this case, execute SET PI 0 or SET PI 1 and cancel the Mode.
Then you can execute the command again.

```
>SET PI 1 ↓
>G ↓
RUN
```

2) Example of setting watch dog timer

```
>SET WD 1 ↓
>SET ↓

CPU    GMS30140
ROM   1024    Byte            RAM    32   Nibble
I/O  Input  [pi]     = [0]  Port Input I/O register
I/O  Output [po]     = [0]  No Display
Symbol      [1]      = [0]  Search
Watch Dog timer [wd] = [1]  ON (no option)
>
```

SL (Symbol file Load)

[Function]
To load the symbol table from the specified symbol file.

[Format]
SL_<file name>↓

[Explanation]
The system reads the symbol table from the specified symbol file.
When symbols are used with `U` or `A` or other commands on this system, this command should be executed first prior to the execution of those commands.
When executing this command, the symbol table in the memory shall be cleared.
Data not in the set format shall not be loaded.
File format : Address_symbol_I ↓
(_indicates space or tab. Space or tab after the symbol is valid but those coming later shall be ignored.)
Address should be polynomial here.

[e.g]

```
>SL_TEST. CRF ↓
. . . . . .
Symbol load OK!
>
```

ST (Status)

[Function]
To display the internal set condition of the simulator.

[Format]
ST↓

[Explanation]
The status of the simulator shall be displayed.

[e.g]

```
>ST ↓
CPU    GMS30140
ROM   1024    Byte           RAM    32   Nibble
I/O  Input  [pi]             = [0]  Port input I/O register
I/O  Output [po]             = [0]  No Display
Symbol      [1]              = [0]  Search
Watch Dog timer [WD]         = [0]  OFF
PC=00  PA=00  PB=00  A=0  X=0  Y=0  ST=0  PMR=0  WD=000  MC=0
                     SP=0  SR0=0000  SR1=0000  SR2=0000
I/O Reg.  Rin=1111  Rout=1111  D=00000000  OUT=0  K=0000(0h)
>
```

T (Trace)

[Function]

To execute the program in the specified program memory address in a single step.

[Format]

T_[=<address>] [_<step>]↓

[Explanation]

With this, the system executes the program in the specified program memory address by a single step.
This command shall be valid until `.` key is pressed. That is, every time `↓` key is pressed after executing this command, the command is executed by one step.
The No. of steps set here is in hexadecimal.

[e.g]

```
>T =F00  2↓
COUNT  =  0000              LPBI                0D
0F00     1B
PC=0F  PA=01  PB=0D  A=0  X=0  Y=0  ST=1  PMR=0  WD=0001  MC=0
COUNT  =  0001
0F01     87                 BR                  07
PC=0D  PA=07  PB=0D  A=0  X=0  Y=0  ST=1  PMR=0  WD=0002  MC=2
. ↓
>
```

TMT (Time calculate)

[Function]
To calculate time from No. of machine cycles and clock frequency.

[Format]
TMT_m_c↓
m : No. of machine cycles
Without any calculating factors.
c : Clock frequency (1K -10m)
Calculating factors must always be input in small letters.
k (kilohertz)
m (megahertz)

[Explanation]
With this, the system calculates time from No. of machine cycles and clock frequency. Range of obtainable time :

6ns - 7158h 16m 43s 770ms

(nano second) (hour) (minute) (second) (millisecond)

Calculating equation : machine cycle × (1/clock frequency × 6)

[e.g]

```
>TMT_1000_1m ↓
6ms   0m   0ns
>
```

TMC (Time calculate)

[Function]
To calculate No. of machine cycles from time and clock frequency.

[Format]
TMC_t_c↓
t : Time
Calculating factors must always be input in small letters.
h (hour)
m (minute)
s (second)
ms (millisecond)
us (microsecond)
ns (nano second)
c : Clock frequency (1K -10m)
Calculating factor must always be input in small letters.
k (kilohertz)
m (megahertz)

[Explanation]
With this, the system calculates No. of machine cycles and clock from time and clock frequency. Range of obtainable machine cycle : 1-14984999833500
Calculating equation : Time ÷ (1/clock frequency × 6)

[e.g]

```
>TMC  6ms   1m↓
MC=1000
>
```

U (Unassemble)

[Function]
To unassemble the area specified with <In> from the specified <address> and to display in mnemonic.

[Format]
U_[<address>] [_<In>]↓

[Explanation]
With this, the system unassembles the area specified with <In> from the specified <address> and displays in mnemonic.
When [<address>] [<In>] are omitted, unassembling is performed from the address immediately after the display start address.
Default value for <In> is 16.
However, [<address>] cannot be omitted separately.

[e.g]

```
>U  200  L4 ↓
0200      40     LYI     00
0201      21     LAM
0203      77     ALEI    0E
0207      AF     BR      2F
>
```

WP (Write Program to MS-DOS* file)

[Function]
To read data from the whole ROM area and write data in the file specified with <file name>.

[Format]
WP_<file name>↓

[Explanation]
With this, the system reads data from the whole ROM area and writes data in the file specified with <file name>.
The file format is the same as the Intel hexa.

[e.g]

```
>WP_test.  rhx ↓
. . . . . .. . . . . .
Program write
```

? (help)

[Function]
To display the list of commands of this simulator.

[Format]
?↓

[Explanation]
With this, the system displays the list of commands of this simulator.

[e.g]
>?↓

```
============================================================
GSEN-GMS30K Simulator Processor is GMS30K Series Version 1.0
============================================================
A[<address>] - assemble                    LOGON - command logging start
BAT <filename> - command repeat            LOGOFF - command logging end
BC [bc] - breakpoint clear                 LP <filename>-load program from PC-DOS
BL -list breakpoint(s)                     MPP <range> <address> - move
BP [bp] <address> - set breakpoint         MPS <range> <address> - move
    [S] <step> - set step breakpoint       P <address> - port input/output
CR - CPU reset                             Q - quit
DD - dump data memory                      R [<reg>] [[=] <value>] - register
DPP [<range>] - dump program memory        SET <value> <range> - simulator setup
DPS [<range>] - dump program memory        SL - <filename> - symbol file load
ED [<address>]-exchange data memory        ST - simulator status dump
EPP[<address>]-exchange program memory     TMT <mc> <clock rate> - machine time
EPS[<address>]-exchange program memory     TMC <time> <clock rate> - step
FD <range> <h> - fill program memory       T [=<address>] [<value>] - trace
FPP <range> <hh> - fill program memory     U [<range>] - unassemble
FPS <range> <hh> - fill program memory     WP <filename> - write program
G [=<address> [<address>..]] - go          ? - help dump
H <value> <value> - hexa add, hexa sub     ^A - command recall
                                           ![DOS command] - shell escape
------------------------------------------------------------
DPP, EPP, FPP, MPP=polynomial address   DPS, EPS, FPS, MPS=sequential address
```

[CTRL]-[A] - command recall

[Explanation]
Repeat the previous command to execute in command prompt>. memorize up to maximum 16 previous commands. That is to say, the previous commands are displayed as many times as [CTRL]-[A] key is pressed.

! [Dos command] - dos shell

[Explanation]
Make it possible to execute the dos command within the simulator environment.

[e.g]

```
>!  dir
  - - -  directory listing  - - -
>
```

File types used in the simulator:

1) Load Module file
2) Input port File (Pseudo Data)
3) BATCH File
4) Log File

Load Module File (RHX File)

Execution File for executing on the simulator.
File format is Intel hexa. format

Input port file (Pseudo data)

When Command concerning to input port is fetched while executing on the simulator, if File Mode has been specified with SET command, data defined in this file shall be read as input data.
(File Name : PORT.DAT)

BATCH File

Each command of the simulator described in 3. shall be consecutively executed according to the order defined in this file.

Log FIle

After executing LOGON command, data displayed on CRT screen shall be written to this file until LOGOFF command is executed.
The file created in this event is the log file.
(File Name : LOG.DAT)

7. Error Message and Troubleshooting

- CRF Error Occurred !

  : In the process of reading Intel Hexa file, it occurs when error happens, Re-assemble the source program and make the Hexa file

- Disk Error !

  : It occurs when disk drive is not prepard.

- File not found !

  : It occurs when the file input by user does not exit in disk. Check the correct filename.

- Help file not found !

  : It occurs when there is no `GMS30K.HLP` file.

- Hexa file format Mismatch !

  : It occurs when the file format of*.rhx file is different. Check if it is intel Hexa format or not.

- Memory not available !

  : It occurs when system memory is lack. Execute the simulator after deleting the memory resident program from system.

- `PORTIN.DAT` File format error!

  : It occurs when the format of PORTIN.DAT file is not correct. Check if it is created correctly according to PORTIN.DAT file structure.

- PORTIN.DAT` File not found !

  : It occurs when there is no PORTIN.DAT file.

- Symbol file format mismatch!

  : It occurs when the format of the symbol file(.CRF) to load is not correct.

- Write error!

  : It occurs when the disk capacity is lack. Store the empty disk.

- ^???

  : It occurs when the commands is input incorrectly in command window. Check if it is the command provided from the simulator.

- ???^

  : It occurs when the command is input correctly in command window, but input format of paramater value is not mismatch. Check the parameter format the corresponding command requests.

- ???

  : It is displayed when the errors except for ^??? and ???^ happen in command window.

INTRODUCTION	1
ARCHITECTURE	2
INSTRUCTION	3
EVALUATION BOARD	4
SOFTWARE	5
APPENDIX	6

Product Change note

Device	Subject
Magic-III	Production Drop
Magic-IIIa	M3004 + SAA3010
Magic-IIIb	uPD6122-001
Magic-IIIc	uPD6122-002

Appendix A Magic- I

Software Spec Sheet

- **Product** 4 bit 1 chip microcomputer
- **Type** Magic- I
- **Purpose** Infrared remote control encoder
- **Function** 8 ×4 key matrix scan
- **Features**
  - Program memory (On-chip ROM) : 1024 bytes
  - Data memory (On-chip RAM) : 32× 4 bits
  - 43 types of instruction sets
  - 3 levels of subroutine nesting
  - 1 bit output port for a large current (Output signal)
  - Operating frequency : 300KHz to 1MHz
  - Instruction cycle : 13.1868μs (fosc=455KHz)
  - CMOS process (Single 3V power supply)
  - Stand-by function (Through internal instruction)
  - Released stand-by mode by key input (Masked option)
  - Built in capacitor for ceramic oscillation circuit (Masked option)
  - Built in a watch dog timer (WDT)
  - 3 kind of double action key for uPD6121G
  - Provided with indicator for transmission

1. Function

1) Stand-by

Output of D0~D3 is ″L″
Oscillation is stop
Output of D4~D5 are held to ″H″

2) Condition of stand-by mode

After reset
When scan strobe output is over, there is not any key input.

3) Release of stand-by mode

One of the key scan input (K0~K3, R0~R3) is changed to ″L″ level.

2. Key Input

1) Key-data mapping table

Output pin	Input pin							
	K0	K1	K2	K3	R0	R1	R2	R3
D0	1	2	3	4	5	6	7	8
D1	9	10	11	12	13	14	15	16
D2	17	18	19	20	21	22	23	24
D3	25	26	27	28	29	30	31	32

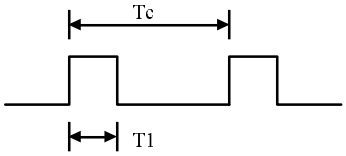
2) Double action key

Combi of Key	Key Data
K21 + K22	35h
K21 + K23	36h
K21 + K24	37h

3. Output

1) Output waveform for uPD6121G

A single pulse, modulated with 37.917KHz signal at 455KHz



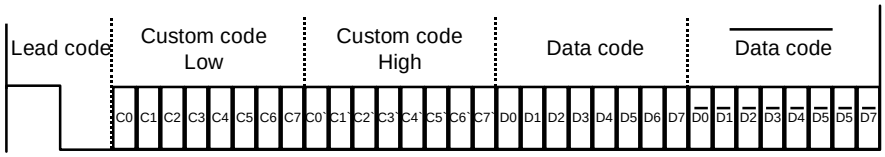
Carrier frequency

$f_{CAR} = 1/Tc = f_{OSC}/12$

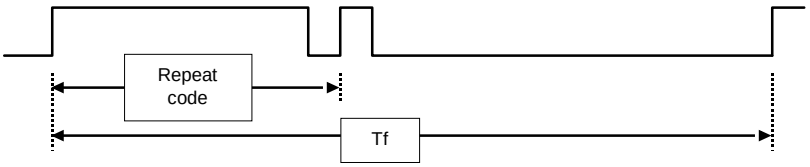
Duty ratio = T1/Tc = 1/3

- Configuration of Flame

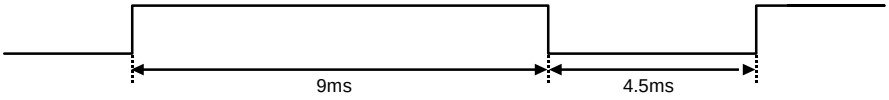
1st flame



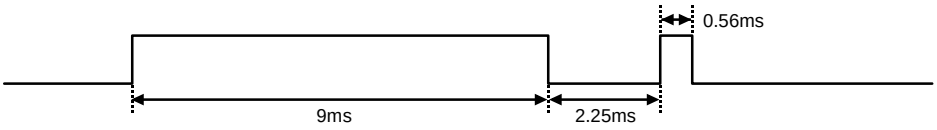
Repeat flame



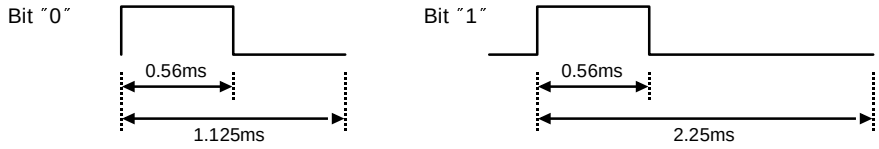
- Lead code



- Repeat code

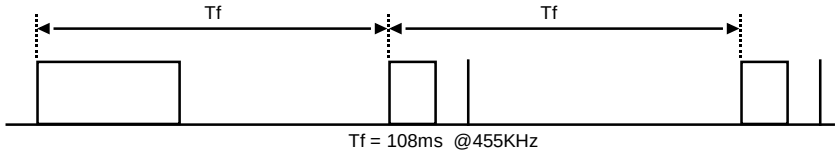


- Bit Description



- Flame Interval : Tf

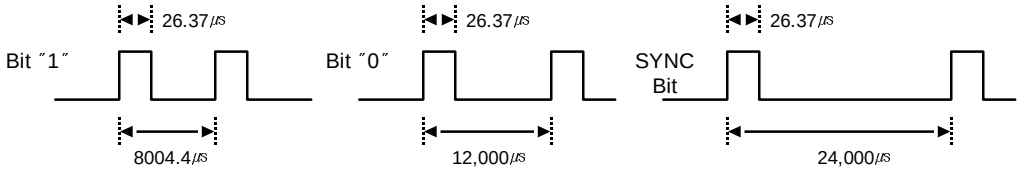
The transmitted waveform as long as a key is depressed



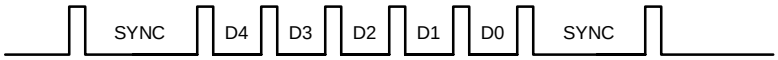
2) Output waveform for MV500

A single pulse at 455KHz

- Bit Description

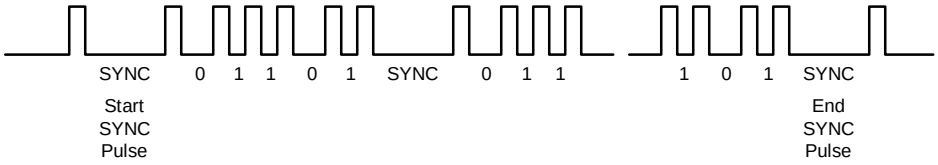


- Configuration of Flame



- Flame Interval : Tf

The transmitted waveform as long as a key is depressed

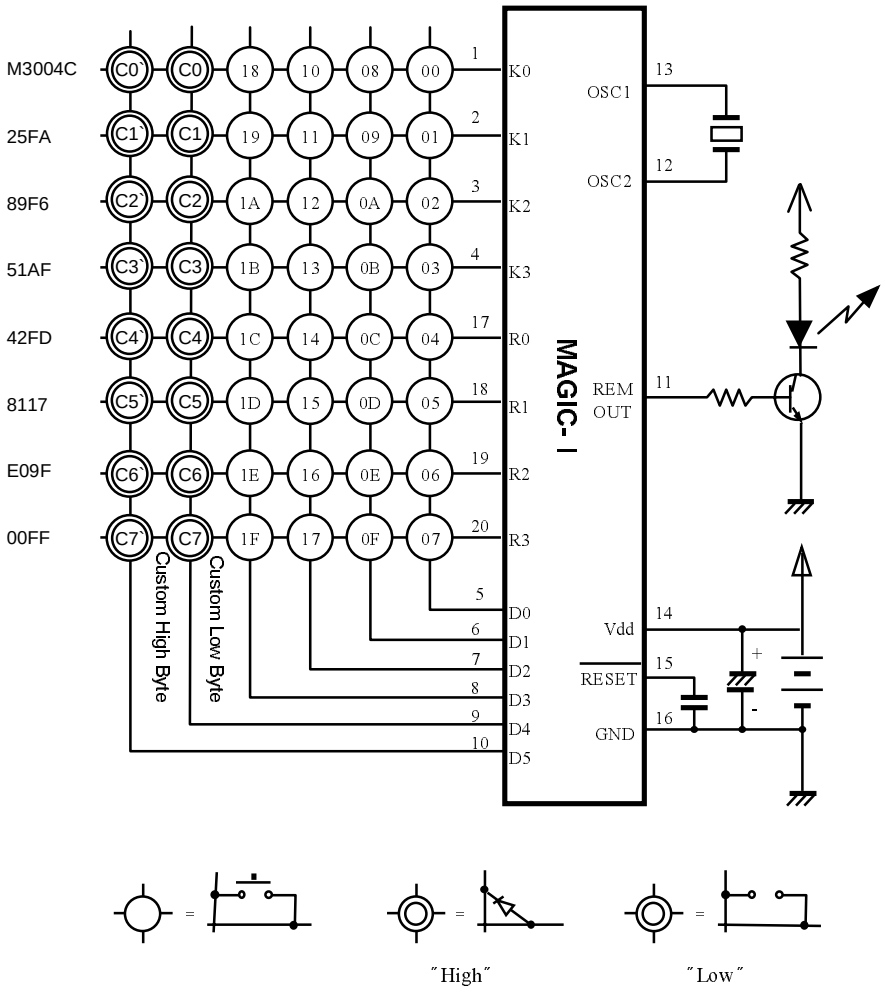


4. Pin Description

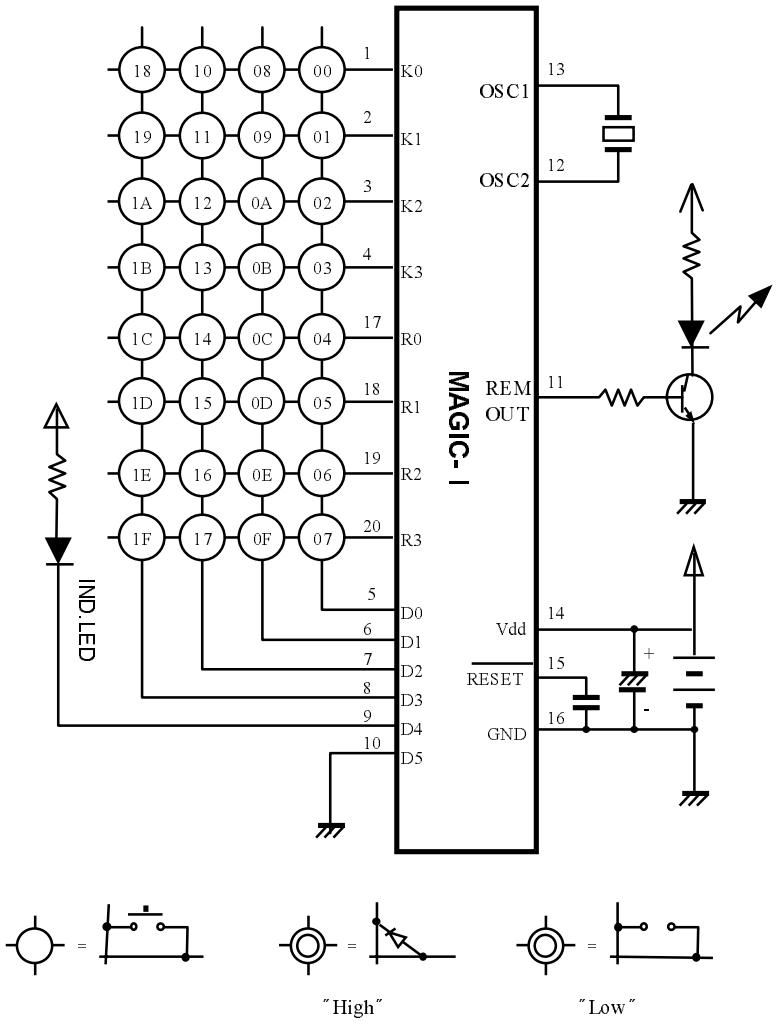
Pin No.	Pin	Funcrion	Remark
1	K0	Key Input No. 0 pin	
2	K1	Key Input No. 1 pin	
3	K2	Key Input No. 2 pin	
4	K3	Key Input No. 3 pin	
5	D0	Key Scan Signal No. 0 pin	
6	D1	Key Scan Signal No. 1 pin	
7	D2	Key Scan Signal No. 2 pin	
8	D3	Key Scan Signal No. 3 pin	
9	D4	Custom code Low input pin	
10	D5	Custom code High input pin	
11	REMOUT	Out of Transmission pin	
12	OSC2	Connect Ceramic Resonator between OSC1 & OSC2	
13	OSC1	Connect Ceramic Resonator between OSC1 & OSC2	
14	V_{DD}	Connect 3V power source	
15	$\overline{\text{RESET}}$	Reset by input of ″L″	
16	GND	Reference voltage for all inputs & outputs 0V	
17	R0	Key Input No. 4 pin	
18	R1	Key Input No. 5 pin	
19	R2	Key Input No. 6 pin	
20	R3	Key Input No. 7 pin	

5. Circuit Diagram & Key Data

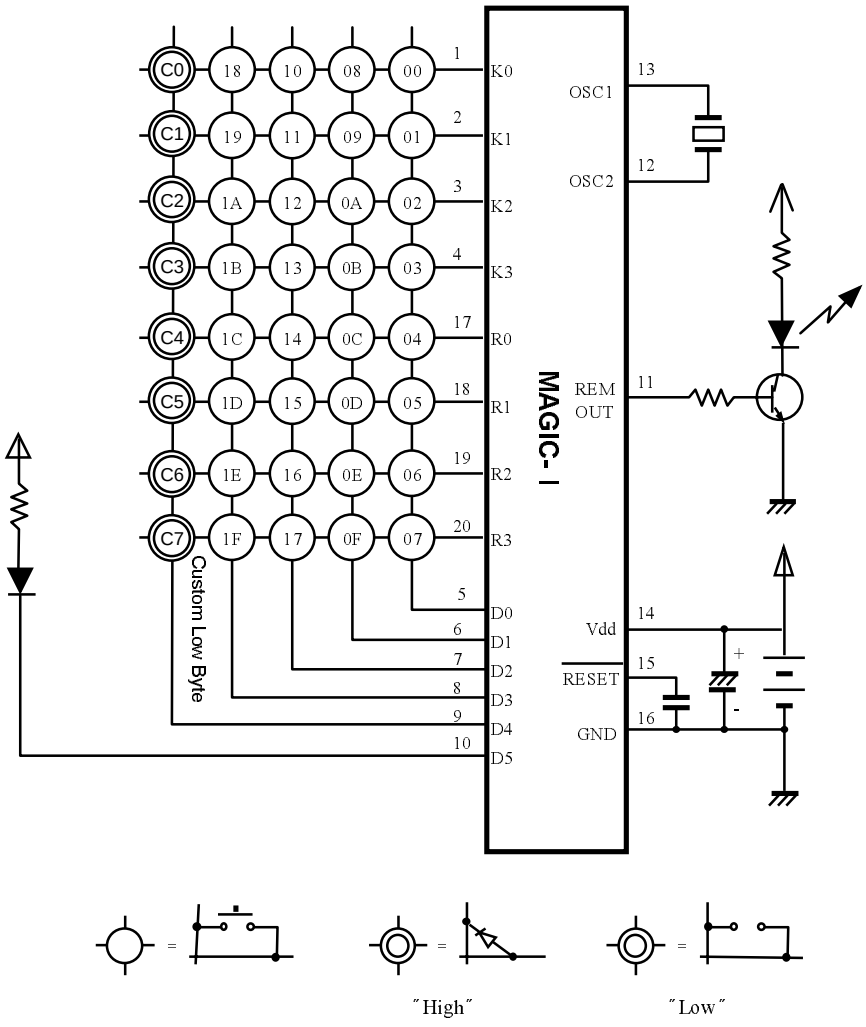
1) NEC 16bit for uPD6121G



2) MV500


18
10
08
00
19
11
09
01
1A
12
0A
02
1B
13
0B
03
1C
14
0C
04
1D
15
0D
05
1E
16
0E
06
1F
17
0F
07
1
2
3
4
17
18
19
20
5
6
7
8
9
10
K0
K1
K2
K3
R0
R1
R2
R3
D0
D1
D2
D3
D4
D5
MAGIC- I
OSC1
OSC2
REM
OUT
Vdd
RESET
GND
13
12
11
14
15
16
IND.LED
+
-
"High"
"Low"

3) NEC 8bit for uPD6121G



Appendix B Magic-Ⅰa

Software Spec Sheet

- **Product** 4 bit 1 chip microcomputer
- **Type** Magic-Ⅰa
- **Purpose** Infrared remote control encoder
- **Function** 8 ×4 key matrix scan
- **Features**
  - Program memory (On-chip ROM) : 512 bytes
  - Data memory (On-chip RAM) : 32× 4 bits
  - 43 types of instruction sets
  - 3 levels of subroutine nesting
  - 1 bit output port for a large current (Output signal)
  - Operating frequency : 300KHz to 1MHz
  - Instruction cycle : 13.1868μs (fosc=455KHz)
  - CMOS process (Single 3V power supply)
  - Stand-by function (Through internal instruction)
  - Released stand-by mode by key input (Masked option)
  - Built in capacitor for ceramic oscillation circuit (Masked option)
  - Built in a watch dog timer (WDT)
  - 3 kind of double action key

1. Function

1) Stand-by

Oscillation is stop

Output of D0~D3 is ″L″, output of D4,D5 is ″H″

2) Condition of stand-by mode

After reset

When scan strobe output is over, there is not any key input.

3) Release of stand-by mode

One of the key scan input (K0~K3, R0~R3) is change to ″L″ level

2. Key Input

1) Key-data mapping table

Output pin	Input pin							
	K0	**K1**	**K2**	**K3**	**R0**	**R1**	**R2**	**R3**
D0	1	2	3	4	5	6	7	8
D1	9	10	11	12	13	14	15	16
D2	17	18	19	20	21	22	23	24
D3	25	26	27	28	29	30	31	32

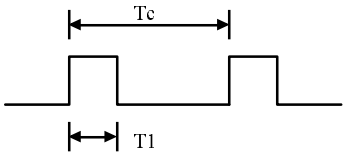
2) Double action key

Combi of Key	Key Data
K21 + K22	35h
K21 + K23	36h
K21 + K24	37h

3. Output

Output waveform for LC7461M-C13

A single pulse, modulated with 37.917KHz signal at 455KHz



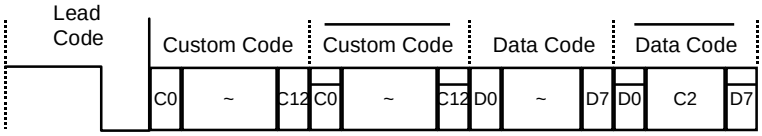
Carrier frequency

$f_{CAR} = 1/Tc = f_{OSC}/12$

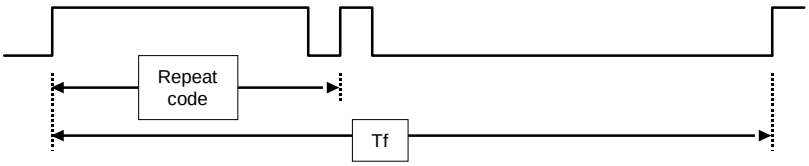
Duty ratio = T1/Tc = 1/3

- Configuration of Flame

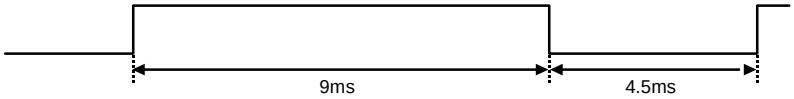
1st flame



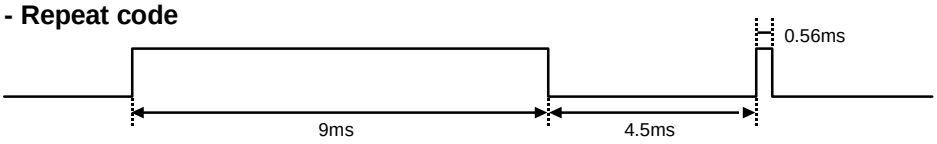
Repeat flame



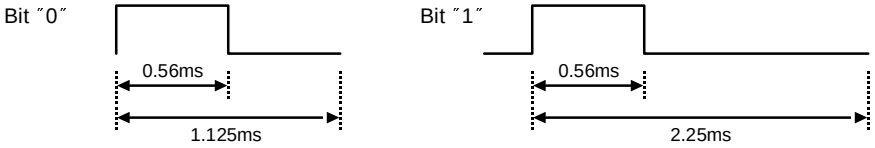
- Lead code



- Repeat code

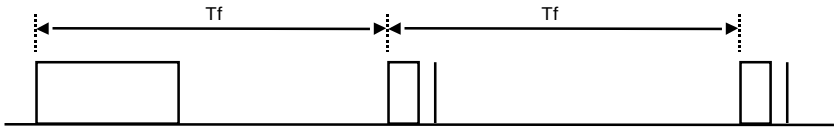


- Bit Description



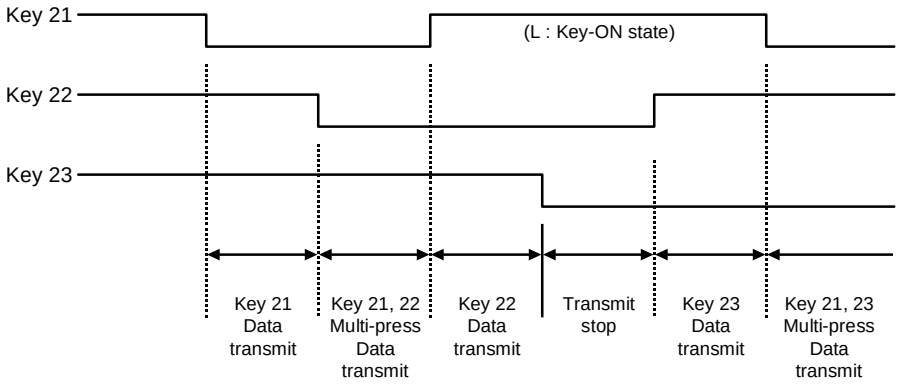
- Flame Interval : Tf

The transmitted waveform as long as a key is depressed



Tf = 108ms @455KHz

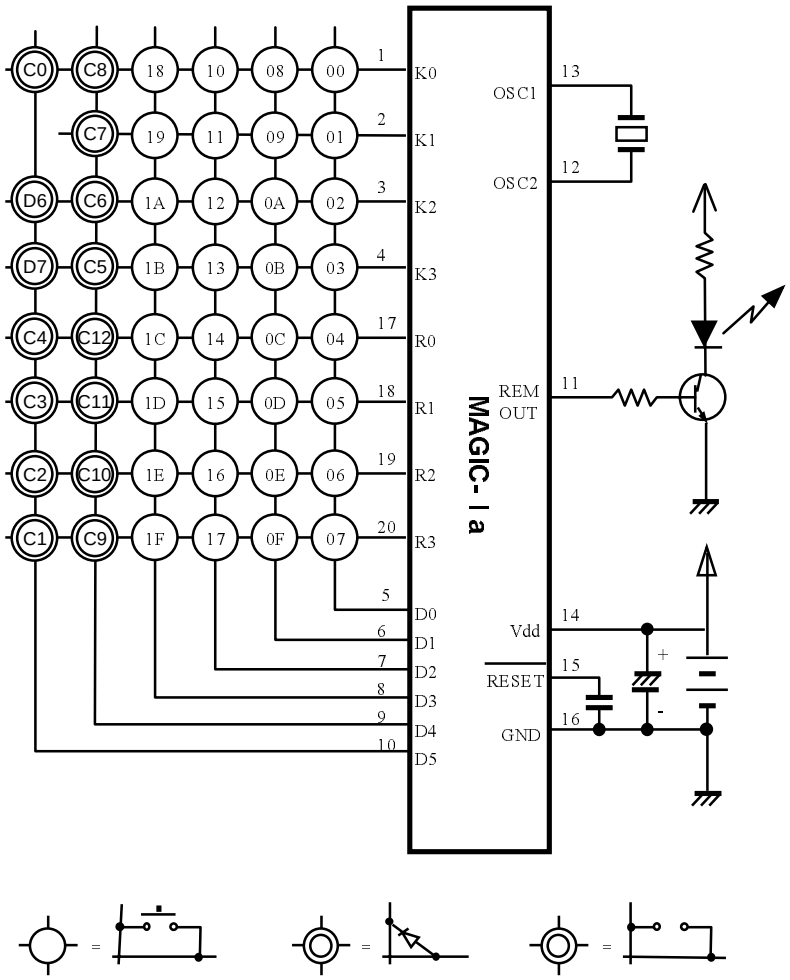
- Double Key Operation



4. Pin Description

Pin No.	Pin	Funcrion	Remark
1	K0	Key Input No. 0 pin	
2	K1	Key Input No. 1 pin	
3	K2	Key Input No. 2 pin	
4	K3	Key Input No. 3 pin	
5	D0	Key Scan Signal No. 0 pin	
6	D1	Key Scan Signal No. 1 pin	
7	D2	Key Scan Signal No. 2 pin	
8	D3	Key Scan Signal No. 3 pin	
9	D4	Custom code input pin	
10	D5	Custom code input pin	
11	REMOUT	Out of Transmission pin	
12	OSC2	Connect Ceramic Resonator between OSC1 & OSC2	
13	OSC1	Connect Ceramic Resonator between OSC1 & OSC2	
14	V_{DD}	Connect 3V power source	
15	$\overline{\text{RESET}}$	Reset by input of ″L″	
16	GND	Reference voltage for all inputs & outputs 0V	
17	R0	Key Input No. 4 pin	
18	R1	Key Input No. 5 pin	
19	R2	Key Input No. 6 pin	
20	R3	Key Input No. 7 pin	

5. Circuit Diagram & Key Data



Appendix C Magic- II

Software Spec Sheet

- **Product** 4 bit 1 chip microcomputer
- **Type** Magic- II
- **Purpose** Infrared remote control encoder
- **Function** 8 ×4 key matrix scan
- **Features**
  - Program memory (On-chip ROM) : 1024 bytes
  - Data memory (On-chip RAM) : 32× 4 bits
  - 43 types of instruction sets
  - 3 levels of subroutine nesting
  - 1 bit output port for a large current (Output signal)
  - Operating frequency : 300KHz to 1MHz
  - Instruction cycle : 13.1868μs (fosc=455KHz)
  - CMOS process (Single 3V power supply)
  - Stand-by function (Through internal instruction)
  - Released stand-by mode by key input (Masked option)
  - Built in capacitor for ceramic oscillation circuit (Masked option)
  - Built in a watch dog timer (WDT)
  - Double action key is not supported

1. Function

1) Stand-by

Output of D0~D3 is ″L″
Oscillation is stop
Output of D4~D5 are held to ″H″

2) Condition of stand-by mode

After reset
When scan strobe output is over, there is not any key input.

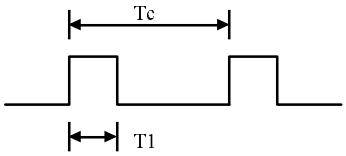
3) Release of stand-by mode

One of the key scan input (K0~K3, R0~R3) is change to ″L″ level

2. Output

1) Output waveform for TC9012F

A single pulse, modulated with 37.917KHz signal at 455KHz



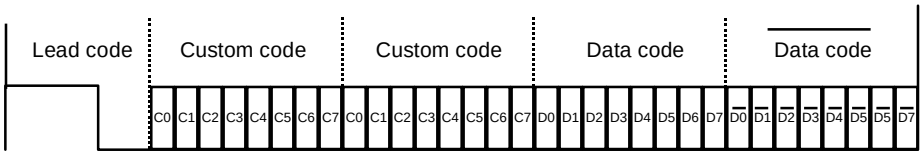
Carrier frequency

$f_{CAR} = 1/Tc = f_{OSC}/12$

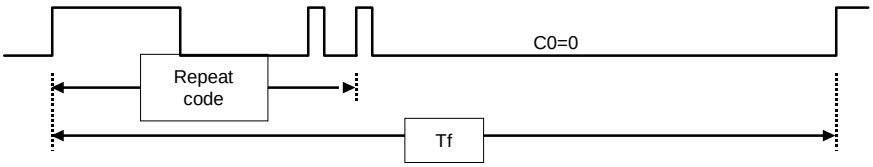
Duty ratio = T1/Tc = 1/3

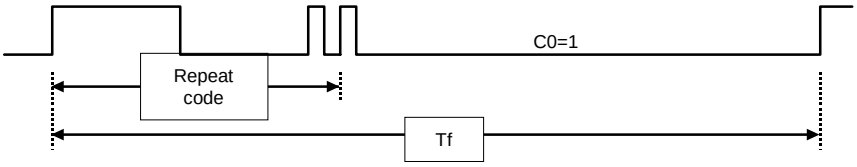
- Configuration of Flame

1st flame

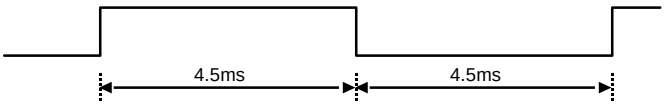


- Repeat flame

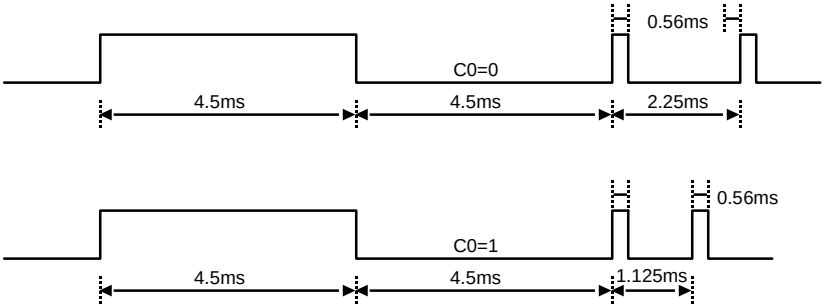




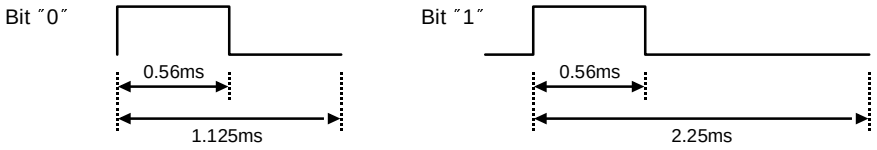
- Lead code



- Repeat code

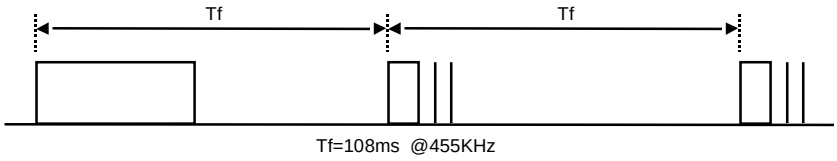


- Bit Description



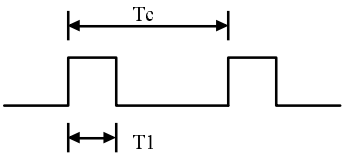
- Flame Interval : Tf

The transmitted waveform as long as a key is depressed



2) Output waveform for uPD1986C

A single pulse, modulated with 37.917KHz signal at 455KHz

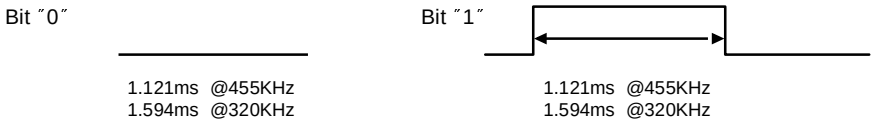


Carrier frequency

$f_{CAR} = 1/Tc = f_{OSC}/12$

Duty ratio = T1/Tc = 1/3

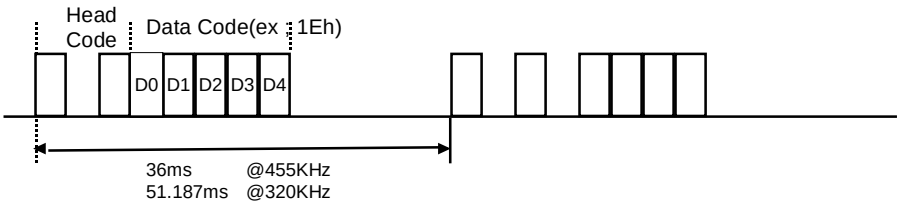
- Bit Description



1.121ms @455KHz
1.594ms @320KHz

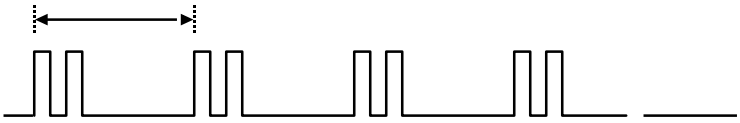
1.121ms @455KHz
1.594ms @320KHz

- Configuration of Flame (Except 18~1Bh Key)



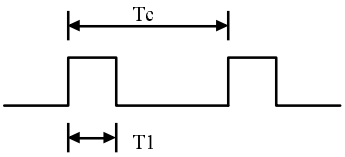
- Flame Interval :,Tf

The transmitted waveform as long as a key is depressed. (18~1Bh Key only)



3) Output waveform for M50560-001P

A single pulse, modulated with 37.917KHz signal at 455KHz

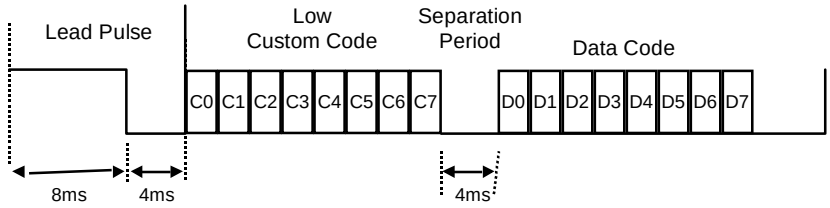


Carrier frequency

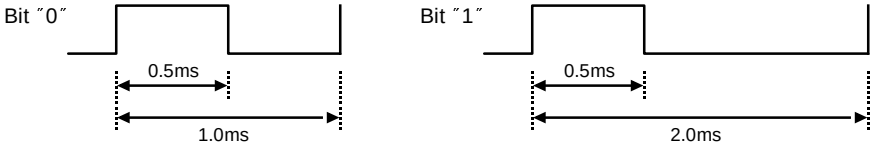
$f_{CAR} = 1/Tc = f_{OSC}/12$

Duty ratio = T1/Tc = 1/3

- Configuration of Flame

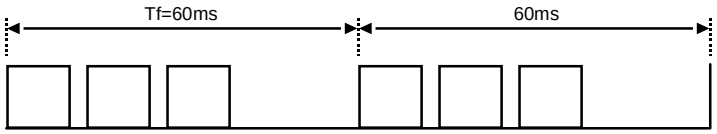


- Bit Description



- Flame Interval :,Tf

The transmitted waveform as long as a key is depressed.

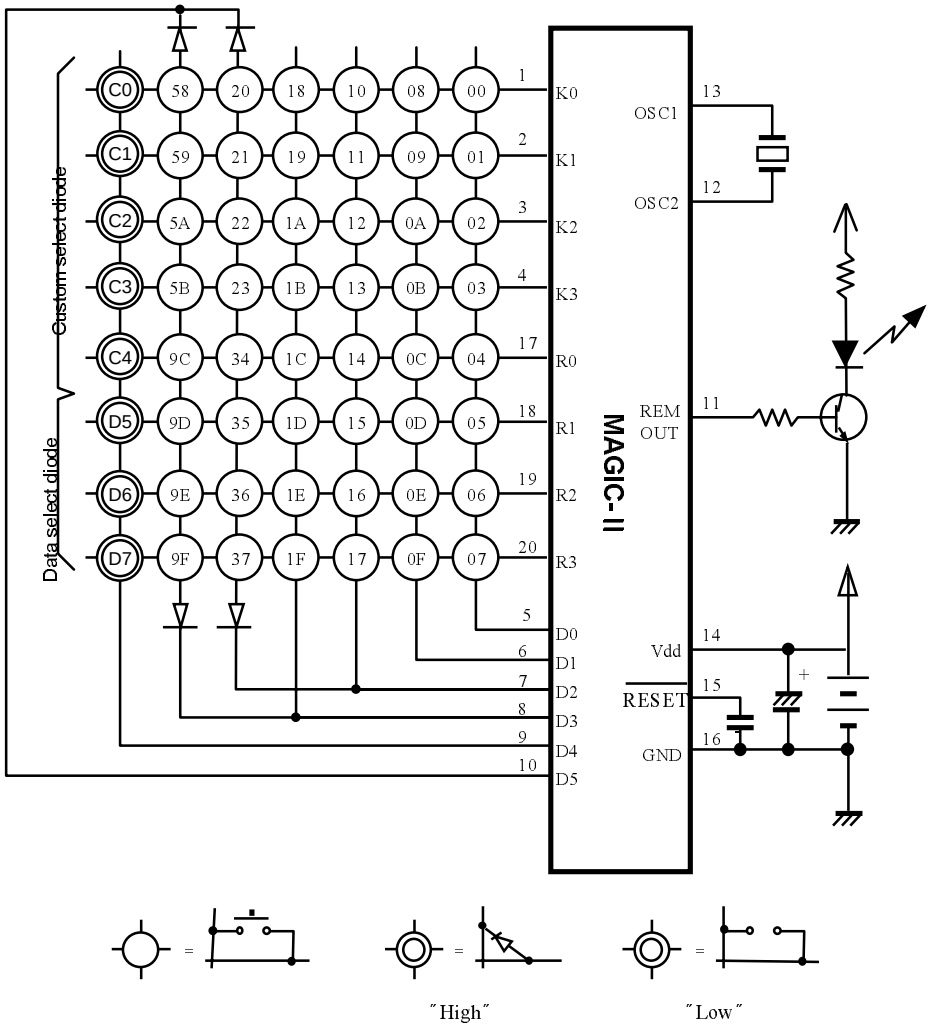


3. Pin Description

Pin No.	Pin	Funcrion	Remark
1	K0	Key Input No. 0 pin	
2	K1	Key Input No. 1 pin	
3	K2	Key Input No. 2 pin	
4	K3	Key Input No. 3 pin	
5	D0	Key Scan Signal No. 0 pin	
6	D1	Key Scan Signal No. 1 pin	
7	D2	Key Scan Signal No. 2 pin	
8	D3	Key Scan Signal No. 3 pin	
9	D4	Custom code input pin	
10	D5	Key Scan Signal No. 4 pin	
11	REMOUT	Out of Transmission pin	
12	OSC2	Connect Ceramic Resonator between OSC1 & OSC2	
13	OSC1	Connect Ceramic Resonator between OSC1 & OSC2	
14	V_{DD}	Connect 3V power source	
15	$\overline{\text{RESET}}$	Reset by input of ″L″	
16	GND	Reference voltage for all inputs & outputs 0V	
17	R0	Key Input No. 4 pin	
18	R1	Key Input No. 5 pin	
19	R2	Key Input No. 6 pin	
20	R3	Key Input No. 7 pin	

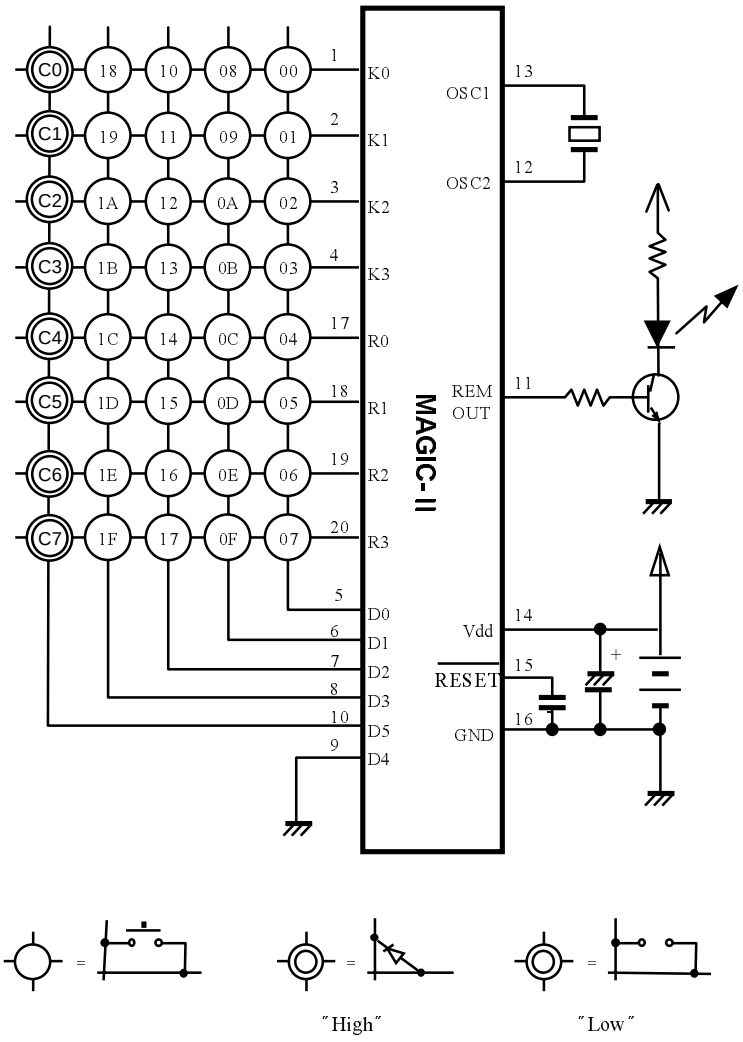
4. Circuit Diagram & Key Data

Mitsubish for M50560-001P

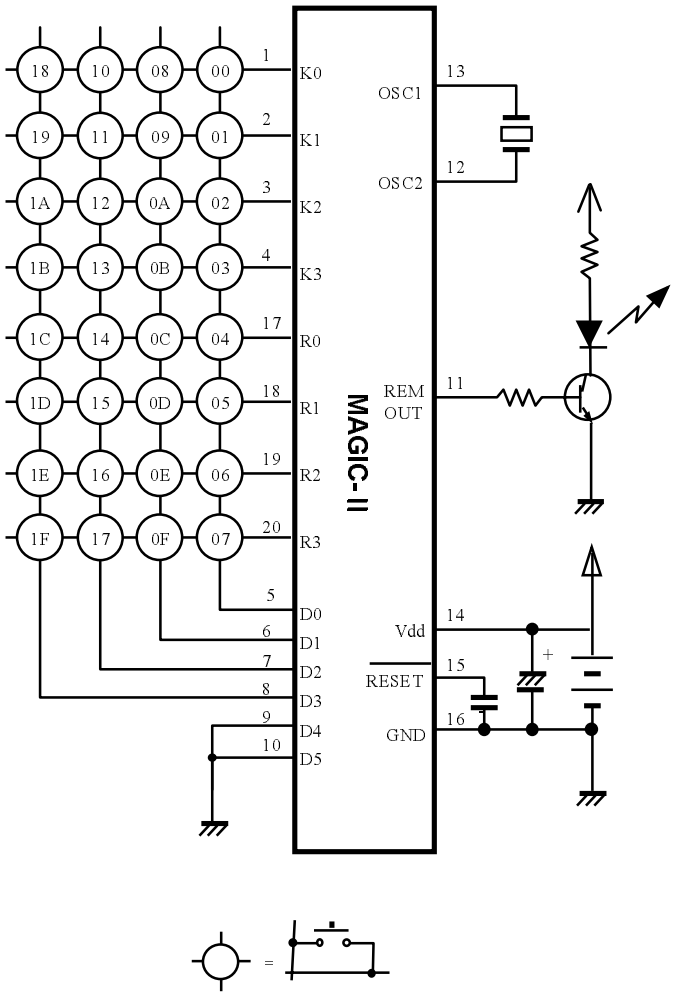


* Extended key data is changed by D5~D7 data selection diode.
This example shows the extended key data when D5~D7 is open.

Toshiba for TC9012



NEC for uPD1986C



Appendix D Magic- II b

Software Spec Sheet

- **Product** 4 bit 1 chip microcomputer
- **Type** Magic- II b
- **Purpose** Infrared remote control encoder
- **Function** 8 ×6 key matrix scan
- **Features**
  - Program memory (On-chip ROM) : 512 bytes
  - Data memory (On-chip RAM) : 32× 4 bits
  - 43 types of instruction sets
  - 3 levels of subroutine nesting
  - 1 bit output port for a large current (Output signal)
  - Operating frequency : 300KHz to 1MHz
  - Instruction cycle : 13.1868μs (fosc=455KHz)
  - CMOS process (Single 3V power supply)
  - Stand-by function (Through internal instruction)
  - Released stand-by mode by key input (Masked option)
  - Built in capacitor for ceramic oscillation circuit (Masked option)
  - Built in a watch dog timer (WDT)
  - Double action key is not supported

1. Function

1) Stand-by

Outputted of D0~D3 is ″L″
Oscillation is stop
Output of D0~D4 is held to ″H″

2) Condition of stand-by mode

After reset
When scan strobe output is over, there is not any key input.

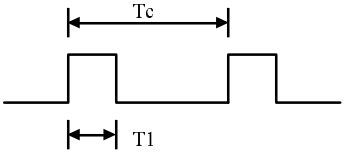
3) Release of stand-by mode

One of the key scan input (K0~K3, R0~R3) is change to ″L″ level

2. Output

waveform for M50560-003P

A single pulse, modulated with 37.91KHz signal at 455KHz

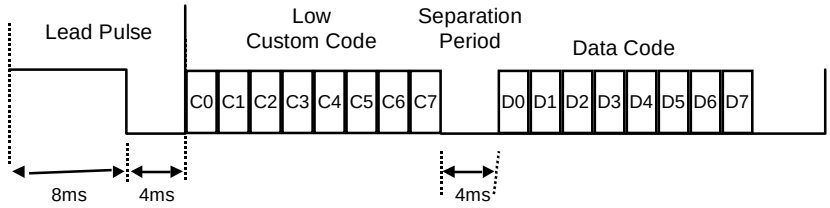


Carrier frequency

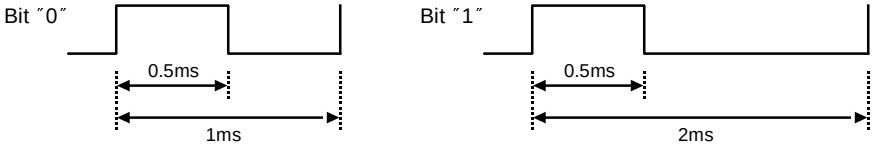
$f_{CAR} = 1/Tc = f_{OSC}/12$

Duty ratio = T1/Tc = 1/3

- Configuration of Flame

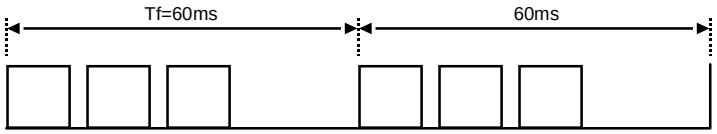


- Bit Description



- Flame Interval :,Tf

The transmitted waveform as long as a key is depressed.



3. Pin Description

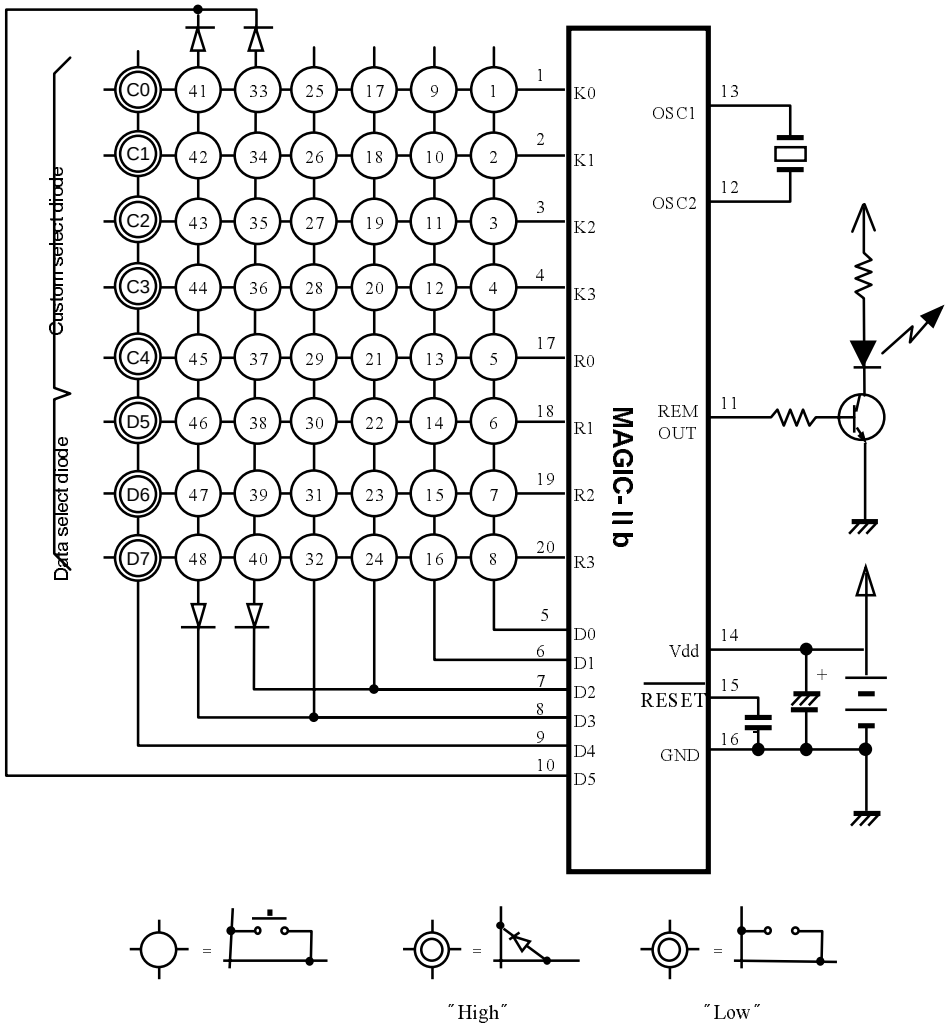
Pin No.	Pin	Funcrion	Remark
1	K0	Key Input No. 0 pin	
2	K1	Key Input No. 1 pin	
3	K2	Key Input No. 2 pin	
4	K3	Key Input No. 3 pin	
5	D0	Key Scan Signal No. 0 pin	
6	D1	Key Scan Signal No. 1 pin	
7	D2	Key Scan Signal No. 2 pin	
8	D3	Key Scan Signal No. 3 pin	
9	D4	Custom code input pin	
10	D5	Key Scan Signal No. 4 pin	
11	REMOUT	Out of Transmission pin	
12	OSC2	Connect Ceramic Resonator between OSC1 & OSC2	
13	OSC1	Connect Ceramic Resonator between OSC1 & OSC2	
14	V_{DD}	Connect 3V power source	
15	$\overline{\text{RESET}}$	Reset by input of ″L″	
16	GND	Reference voltage for all inputs & outputs 0V	
17	R0	Key Input No. 4 pin	
18	R1	Key Input No. 5 pin	
19	R2	Key Input No. 6 pin	
20	R3	Key Input No. 7 pin	

4. Truth Table

Key No.	Key Data(hex) of Diode Select (D5 D6 D7)								Key No.	Key Data(hex) of Diode Select (D5 D6 D7)							
	000	100	010	110	001	101	011	111		000	100	010	110	001	101	011	111
1	00	20	40	60	80	A0	C0	E0	25	18	38	58	78	98	B8	D8	F8
2	01	21	41	61	81	A1	C1	E1	26	19	39	59	79	99	B9	D9	F9
3	02	22	42	62	82	A2	C2	E2	27	1A	3A	5A	7A	9A	BA	DA	FA
4	03	23	43	63	83	A3	C3	E3	28	1B	3B	5B	7B	9B	BB	DB	FB
5	04	24	44	64	84	A4	C4	E4	29	1C	3C	5C	7C	9C	BC	DC	FC
6	05	25	45	65	85	A5	C5	E5	30	1D	3D	5D	7D	9D	BD	DD	FD
7	06	26	46	66	86	A6	C6	E6	31	1E	3E	5E	7E	9E	BE	DE	FE
8	07	27	47	67	87	A7	C7	E7	32	1F	3F	5F	7F	9F	BF	DF	FF
9	08	28	48	68	88	A8	C8	E8	33	20	40	50	80	A0	C0	E0	00
10	09	29	49	69	89	A9	C9	E9	34	21	41	51	81	A1	C1	E1	01
11	0A	2A	4A	6A	8A	AA	CA	EA	35	22	42	52	82	A2	C2	E2	02
12	0B	2B	4B	6B	8B	AB	CB	EB	36	23	43	53	83	A3	C3	E3	03
13	0C	2C	4C	6C	8C	AC	CC	EC	37	34	54	74	94	B4	D4	F4	14
14	0D	2D	4D	6D	8D	AD	CD	ED	38	35	55	75	95	B5	D5	F5	15
15	0E	2E	4E	6E	8E	AE	CE	EE	39	36	56	76	96	B6	D6	F6	16
16	0F	2F	4F	6F	8F	AF	CF	EF	40	37	57	77	97	B7	D7	F7	17
17	10	30	50	70	90	B0	D0	F0	41	58	78	98	B8	D8	F8	18	38
18	11	31	51	71	91	B1	D1	F1	42	59	79	9A	B9	D9	F9	19	39
19	12	32	52	72	92	B2	D2	F2	43	5A	7A	9B	BA	DA	FA	1A	3A
20	13	33	53	73	93	B3	D3	F3	44	5B	7B	9C	BB	DB	FB	1B	3B
21	14	34	54	74	94	B4	D4	F4	45	9C	BC	DC	FC	1C	3C	5C	7C
22	15	35	55	75	95	B5	D5	F5	46	9D	BD	DD	FD	1D	3D	5D	7D
23	16	36	56	76	96	B6	D6	F6	47	9E	BE	DE	FE	1E	3E	5E	7E
24	17	37	57	77	97	B7	D7	F7	48	9F	BF	DF	FF	1F	3F	5F	7F

5. Circuit Diagram

Mitsubish for M50560-003P



Appendix E Magic-Ⅲa

Software Spec Sheet

- **Product** 4 bit 1 chip microcomputer
- **Type** Magic-Ⅲa
- **Purpose** Infrared remote control encoder
- **Function** 8 ×8 key matrix scan
- **Features**
  - Program memory (On-chip ROM) : 1024 bytes
  - Data memory (On-chip RAM) : 32× 4 bits
  - 43 types of instruction sets
  - 3 levels of subroutine nesting
  - 1 bit output port for a large current (Output signal)
  - Operating frequency : 300KHz to 1MHz
  - Instruction cycle : 13.1868μs (fosc=455KHz)
  - CMOS process (Single 3V power supply)
  - Stand-by function (Through internal instruction)
  - Released stand-by mode by key input (Masked option)
  - Built in capacitor for ceramic oscillation circuit (Masked option)
  - Built in a watch dog timer (WDT)
  - Double action key is not supported

1. Function

1) Stand-by

Oscillation is stop

Output of D0~D5, D8, D9 is ″L″, Output of D6~D7 is ″H″

2) Condition of stand-by mode

After reset

When scan strobe output is over, there is not any key input.

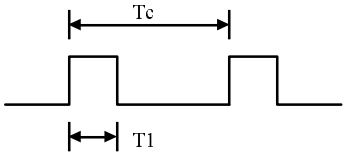
3) Release of stand-by mode

One of the key scan input (K0~K3, R0~R3) is change to ″L″ level

2. Output

1) **Output waveform of M3004C LAB1 : Only Carrier**

A single pulse, modulated with 37.917KHz signal at 455KHz



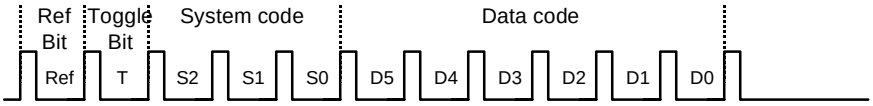
Carrier frequency

$f_{CAR} = 1/Tc = f_{OSC}/12$

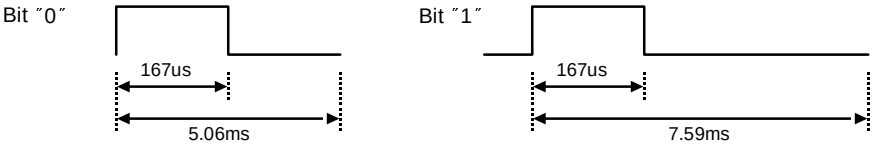
Duty ratio = T1/Tc = 1/3

- Configuration of Flame

1st flame

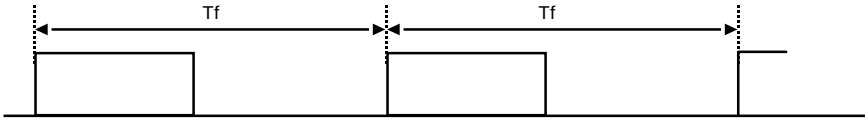


- Bit Description



- Flame Interval : Tf

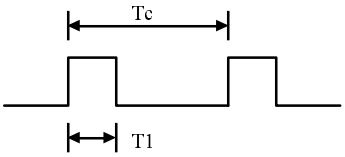
The transmitted waveform as long as a key is depressed



Tf = 121.6ms @455KHz

2) Output waveform of RC-5

A single pulse, modulated with 37.917KHz signal at 455KHz



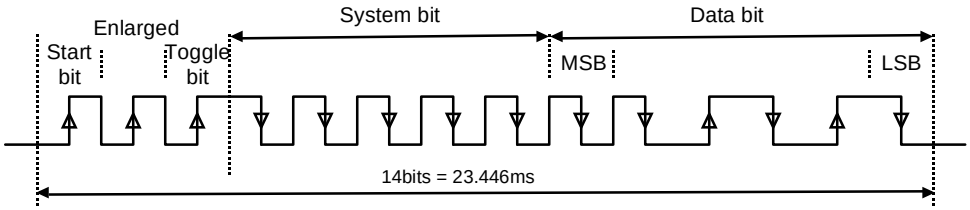
Carrier frequency

$f_{CAR} = 1/Tc = f_{OSC}/12$

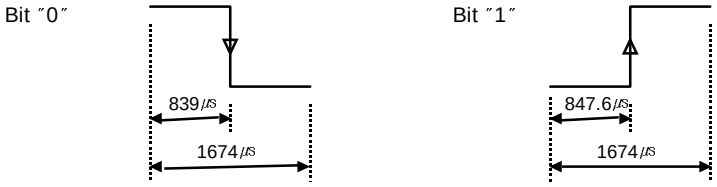
Duty ratio = T1/Tc = 1/3

- Configuration of Flame

1st flame

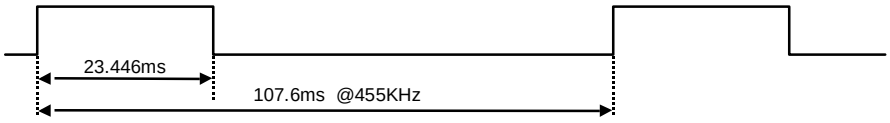


- Bit Description



- Flame Interval : Tf

The transmitted waveform as long as a key is depressed

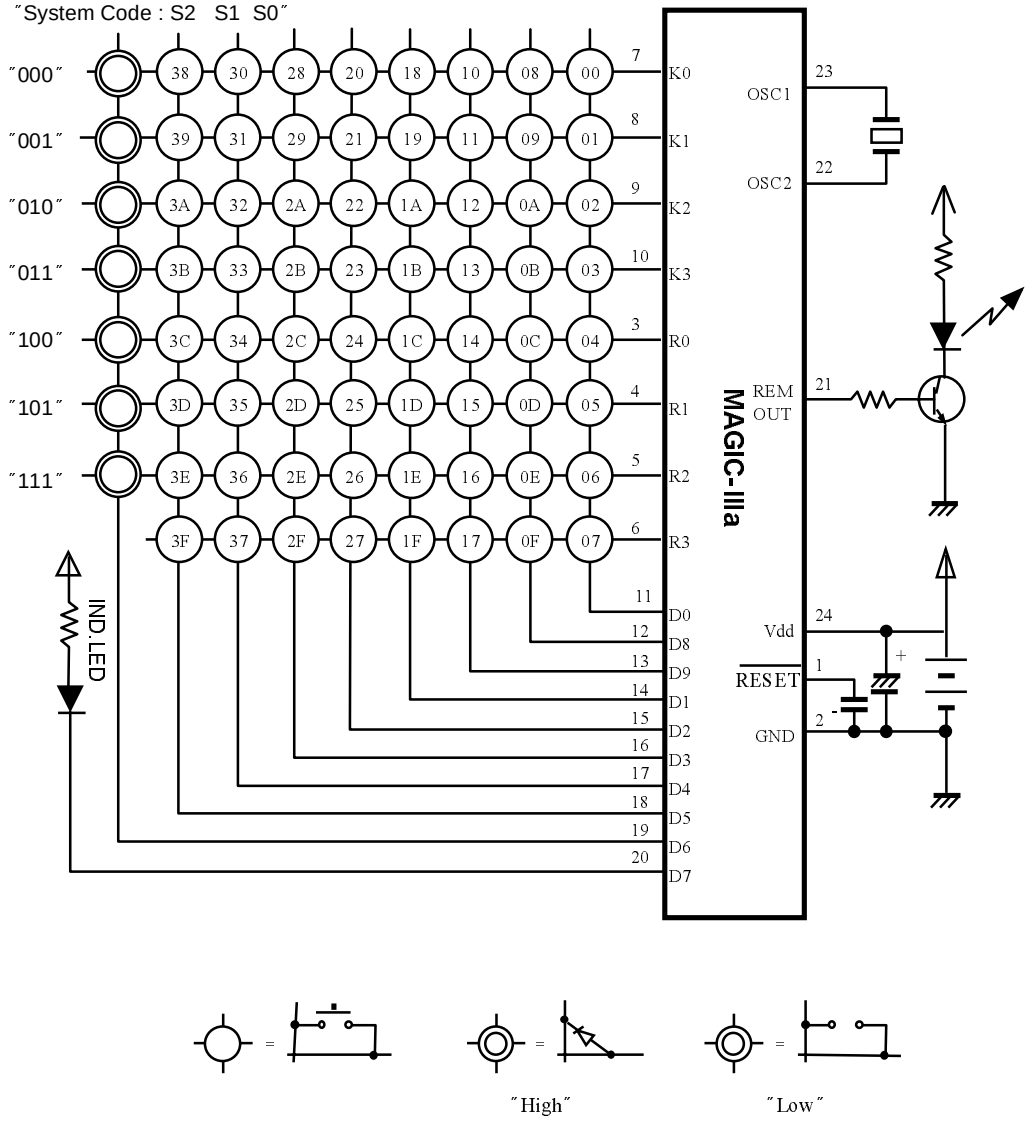


3. Pin Description

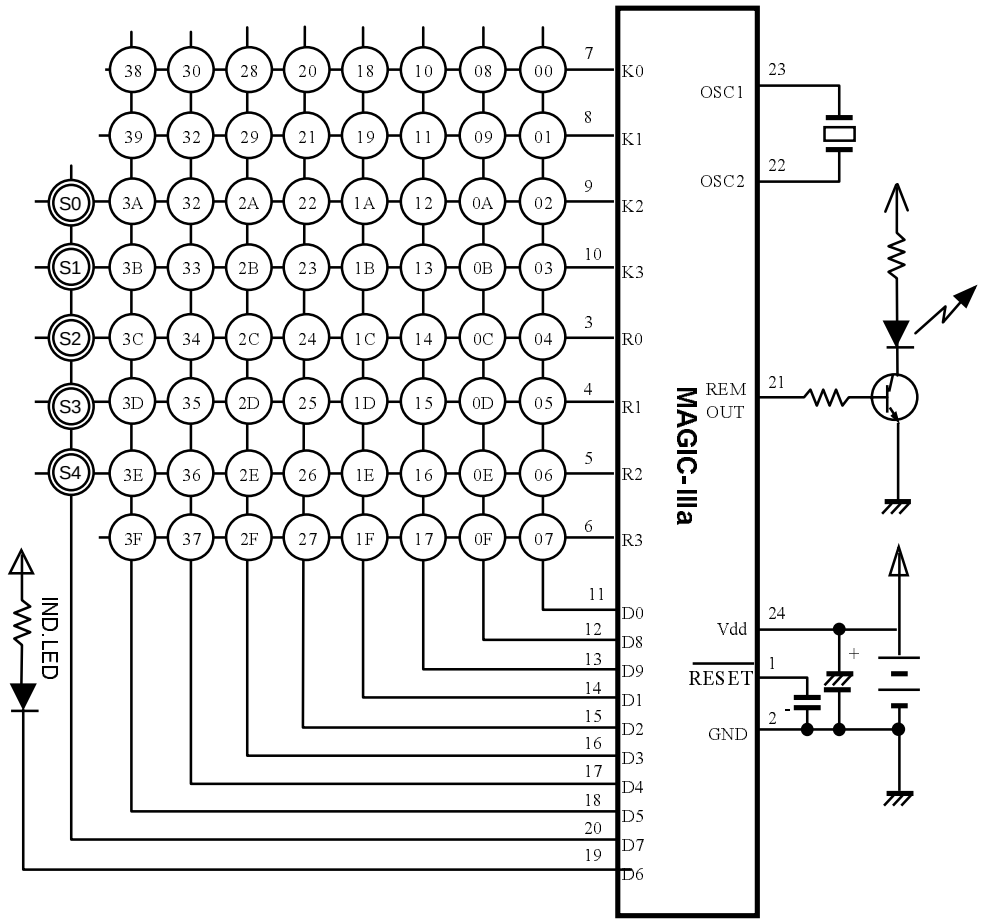
Pin No.	Pin	Funcrion	Remark
1	$\overline{\text{RESET}}$	Reset by input of ″L″	
2	GND	Reference voltage for all inputs & outputs 0V	
3	R0	Key Input No. 4 pin	
4	R1	Key Input No. 5 pin	
5	R2	Key Input No. 6 pin	
6	R3	Key Input No. 7 pin	
7	K0	Key Input No. 0 pin	
8	K1	Key Input No. 1 pin	
9	K2	Key Input No. 2 pin	
10	K3	Key Input No. 3 pin	
11	D0	Key Scan Signal No. 0 pin	
12	D8	Key Scan Signal No. 1 pin	
13	D9	Key Scan Signal No. 2 pin	
14	D1	Key Scan Signal No. 3 pin	
15	D2	Key Scan Signal No. 4 pin	
16	D3	Key Scan Signal No. 5 pin	
17	D4	Key Scan Signal No. 6 pin	
18	D5	Key Scan Signal No. 7 pin	
19	D6	Key Scan Signal No. 8 pin	
20	D7	Key Scan Signal No. 9 pin	
21	REMOUT	Out of Transmission pin	
22	OSC2	Connect Resonator between OSC1 & OSC2	
23	OSC1	Connect Resonator between OSC1 & OSC2	
24	V_{DD}	Connect 3V power source	

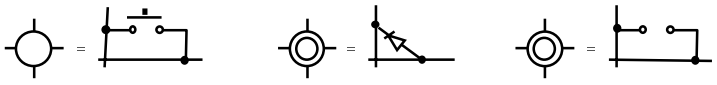
4. Circuit Diagram & Key Data

1) M3004C LAB1



2) Philips for SAA3010





″High″ ″Low″

Appendix F Magic-Ⅲb

Software Spec Sheet

- **Product** 4 bit 1 chip microcomputer
- **Type** Magic-Ⅲb
- **Purpose** Infrared remote control encoder
- **Function** 8 ×8 key matrix scan
- **Features**
  - Program memory (On-chip ROM) : 1024 bytes
  - Data memory (On-chip RAM) : 32× 4 bits
  - 43 types of instruction sets
  - 3 levels of subroutine nesting
  - 1 bit output port for a large current (Output signal)
  - Operating frequency : 300KHz to 1MHz
  - Instruction cycle : 13.1868μs (fosc=455KHz)
  - CMOS process (Single 3V power supply)
  - Stand-by function (Through internal instruction)
  - Released stand-by mode by key input (Masked option)
  - Built in capacitor for ceramic oscillation circuit (Masked option)
  - Built in a watch dog timer (WDT)
  - 3 kind of double action key

1. Function

1) Stand-by

Oscillation is stop

Output of D0~D5, D8, D9 is ″L″, Output of D6~D7 is ″H″

2) Condition of stand-by mode

After reset

When scan strobe output is over, there is not any key input.

3) Release of stand-by mode

One of the key scan input (K0~K3, R0~R3) is change to ″L″ level

2. Key Input

1) Key-data mapping table

Output pin	Input pin							
	K0	**K1**	**K2**	**K3**	**R0**	**R1**	**R2**	**R3**
D0	1	2	3	4	5	6	7	8
D8	9	10	11	12	13	14	15	16
D9	17	18	19	20	21	22	23	24
D1	25	26	27	28	29	30	31	32
D2	33	34	35	36	37	38	39	40
D3	41	42	43	44	45	46	47	48
D4	49	50	51	52	53	54	55	56
D5	57	58	59	60	61	62	63	64

2) Double action key

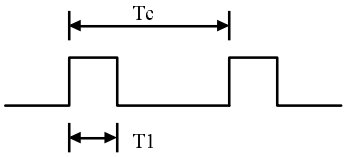
Combi of Key	Key Data
K21 + K22	35h
K21 + K23	36h
K21 + K24	37h

2. Output

Output waveform for uPD6122-001

- Output waveform

A single pulse, modulated with 37.917KHz signal at 455KHz



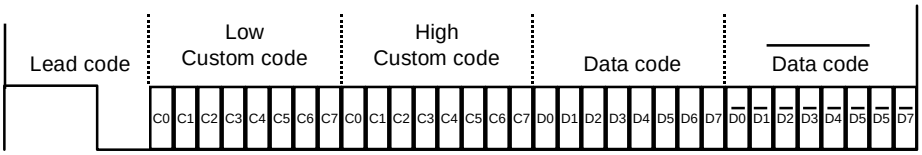
Carrier frequency

$f_{CAR} = 1/Tc = f_{OSC}/12$

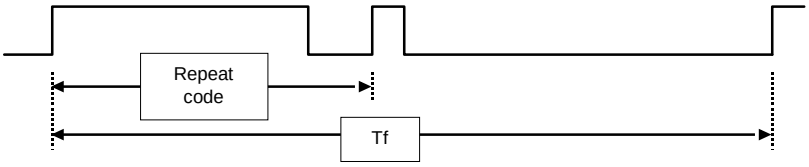
Duty ratio = T1/Tc = 1/3

- Configuration of Flame

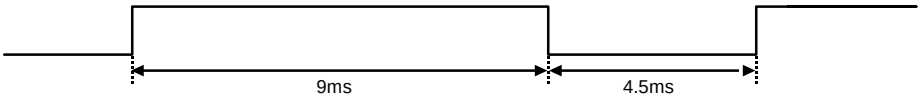
1st flame



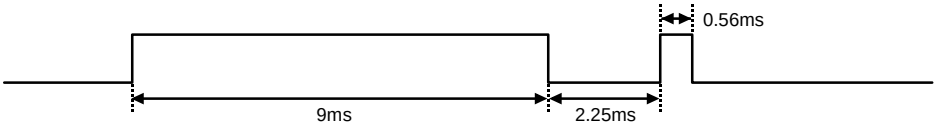
Repeat flame



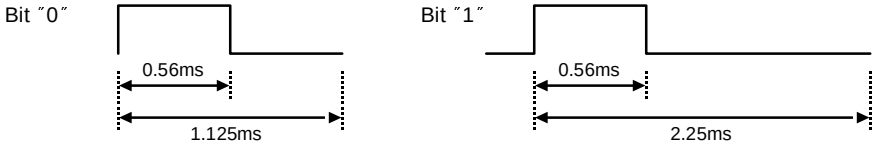
- Lead code



- Repeat code

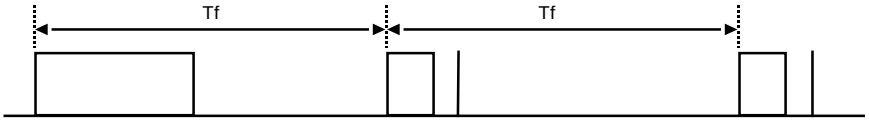


- Bit Description



- Flame Interval : Tf

The transmitted waveform as long as a key is depressed



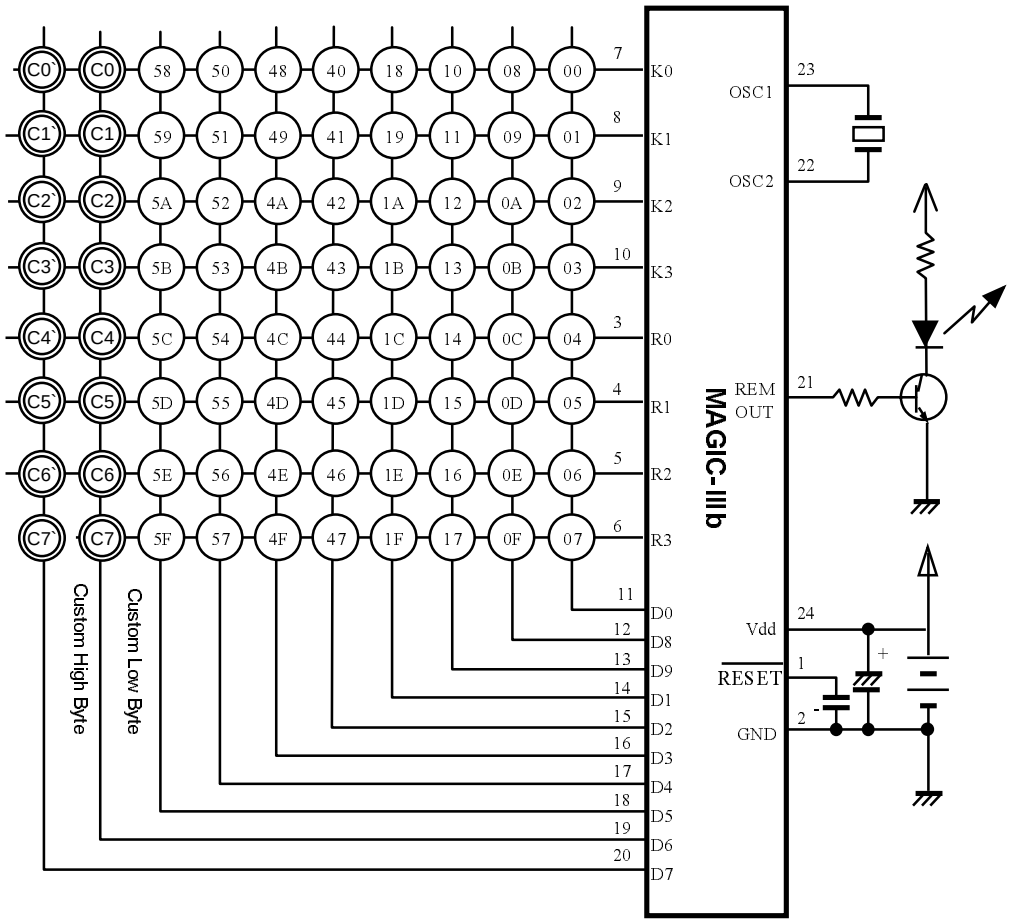
Tf = 108ms @455KHz

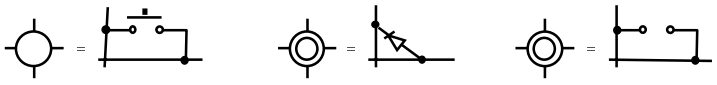
3. Pin Description

Pin No.	Pin	Funcrion	Remark
1	$\overline{\text{RESET}}$	Reset by input of ″L″	
2	GND	Reference voltage for all inputs & outputs 0V	
3	R0	Key Input No. 4 pin	
4	R1	Key Input No. 5 pin	
5	R2	Key Input No. 6 pin	
6	R3	Key Input No. 7 pin	
7	K0	Key Input No. 0 pin	
8	K1	Key Input No. 1 pin	
9	K2	Key Input No. 2 pin	
10	K3	Key Input No. 3 pin	
11	D0	Key Scan Signal No. 0 pin	
12	D8	Key Scan Signal No. 1 pin	
13	D9	Key Scan Signal No. 2 pin	
14	D1	Key Scan Signal No. 3 pin	
15	D2	Key Scan Signal No. 4 pin	
16	D3	Key Scan Signal No. 5 pin	
17	D4	Key Scan Signal No. 6 pin	
18	D5	Key Scan Signal No. 7 pin	
19	D6	Key Scan Signal No. 8 pin	
20	D7	Key Scan Signal No. 9 pin	
21	REMOUT	Out of Transmission pin	
22	OSC2	Connect Resonator between OSC1 & OSC2	
23	OSC1	Connect Resonator between OSC1 & OSC2	
24	V_{DD}	Connect 3V power source	

4. Circuit Diagram & Key Data

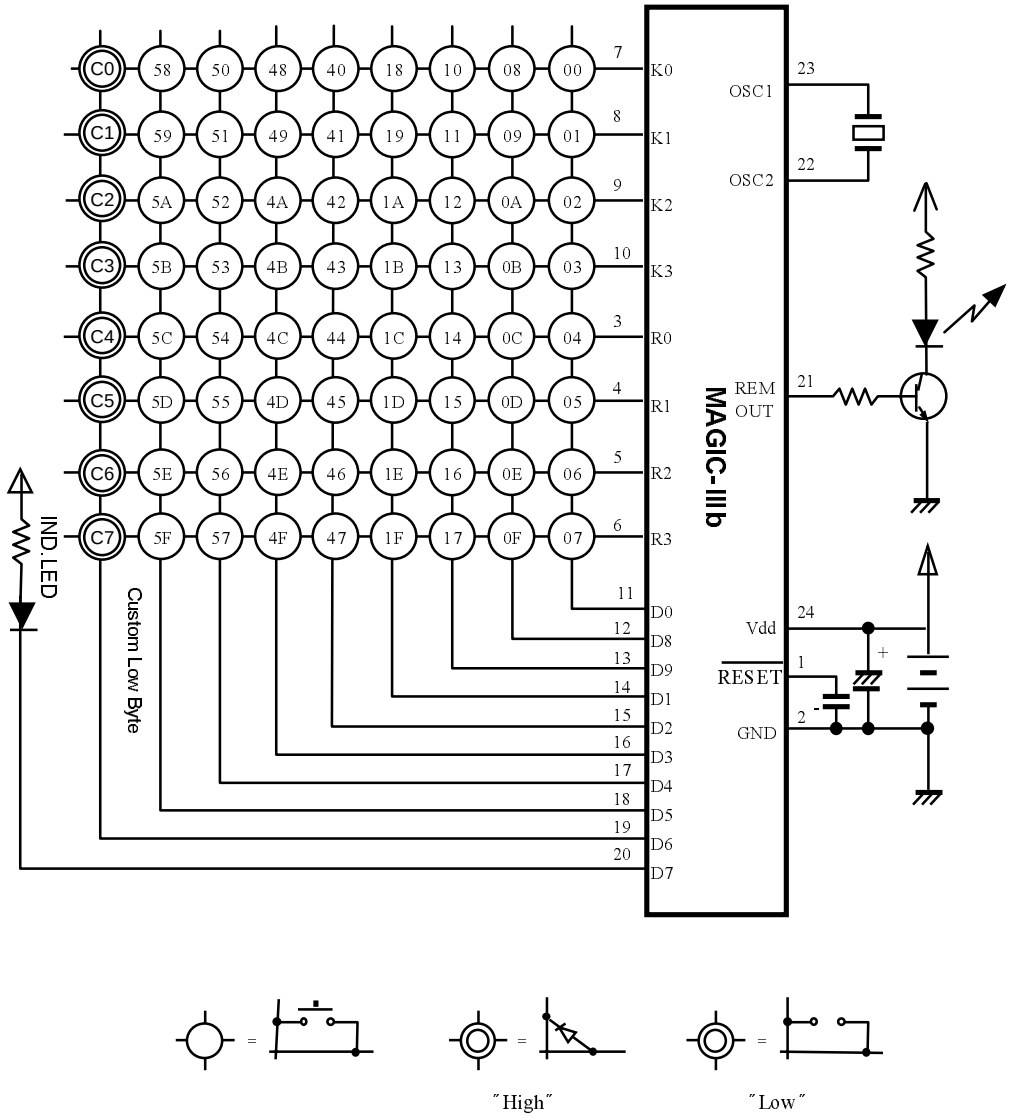
1) NEC 16bit for uPD6122-001





″High″ ″Low″

2) NEC 8bit for uPD6122-001



″High″ ″Low″

Appendix G Magic-Ⅲc

Software Spec Sheet

- **Product** 4 bit 1 chip microcomputer
- **Type** Magic-Ⅲc
- **Purpose** Infrared remote control encoder
- **Function** 8 ×8 key matrix scan
- **Features**
  - Program memory (On-chip ROM) : 1024 bytes
  - Data memory (On-chip RAM) : 32× 4 bits
  - 43 types of instruction sets
  - 3 levels of subroutine nesting
  - 1 bit output port for a large current (Output signal)
  - Operating frequency : 300KHz to 1MHz
  - Instruction cycle : 13.1868μs (fosc=455KHz)
  - CMOS process (Single 3V power supply)
  - Stand-by function (Through internal instruction)
  - Released stand-by mode by key input (Masked option)
  - Built in capacitor for ceramic oscillation circuit (Masked option)
  - Built in a watch dog timer (WDT)
  - 3 kind of double action key

1. Function

1) Stand-by

Oscillation is stop

Output of D0~D5, D8, D9 is ″L″, Output of D6~D7 is ″H″

2) Condition of stand-by mode

After reset

When scan strobe output is over, there is not any key input.

3) Release of stand-by mode

One of the key scan input (K0~K3, R0~R3) is change to ″L″ level

2. Key Input

1) Key-data mapping table

Output pin	Input pin							
	K0	**K1**	**K2**	**K3**	**R0**	**R1**	**R2**	**R3**
D0	1	2	3	4	5	6	7	8
D8	9	10	11	12	13	14	15	16
D9	17	18	19	20	21	22	23	24
D1	25	26	27	28	29	30	31	32
D2	33	34	35	36	37	38	39	40
D3	41	42	43	44	45	46	47	48
D4	49	50	51	52	53	54	55	56
D5	57	58	59	60	61	62	63	64

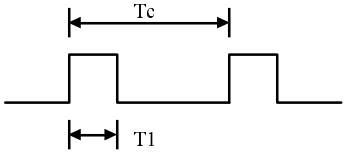
2) Double action key

Combi of Key	Key Data
K21 + K22	B5h
K21 + K23	B6h
K21 + K24	B7h

2. Output

Output waveform for uPD6122-002

A single pulse, modulated with 37.917KHz signal at 455KHz



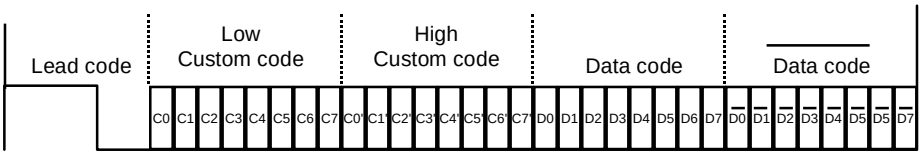
Carrier frequency

$f_{CAR} = 1/Tc = f_{OSC}/12$

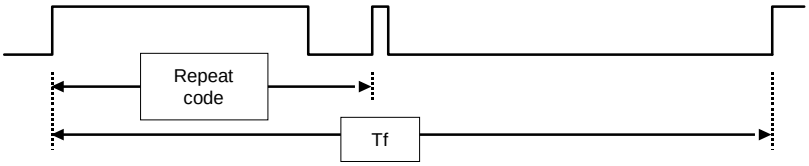
Duty ratio = T1/Tc = 1/3

- Configuration of Flame

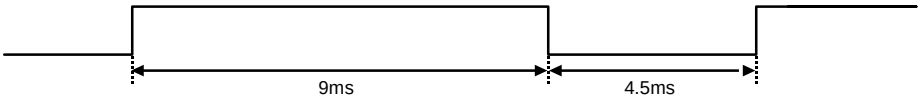
1st flame



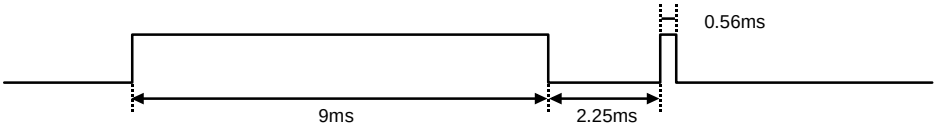
Repeat flame



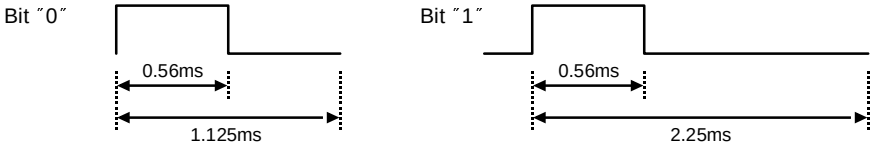
- Lead code



- Repeat code

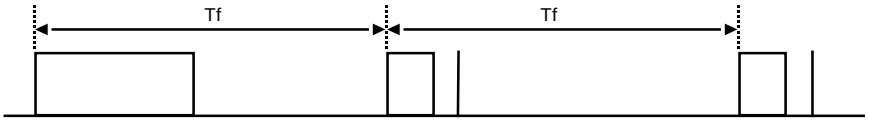


- Bit Description



- Flame Interval : Tf

The transmitted waveform as long as a key is depressed



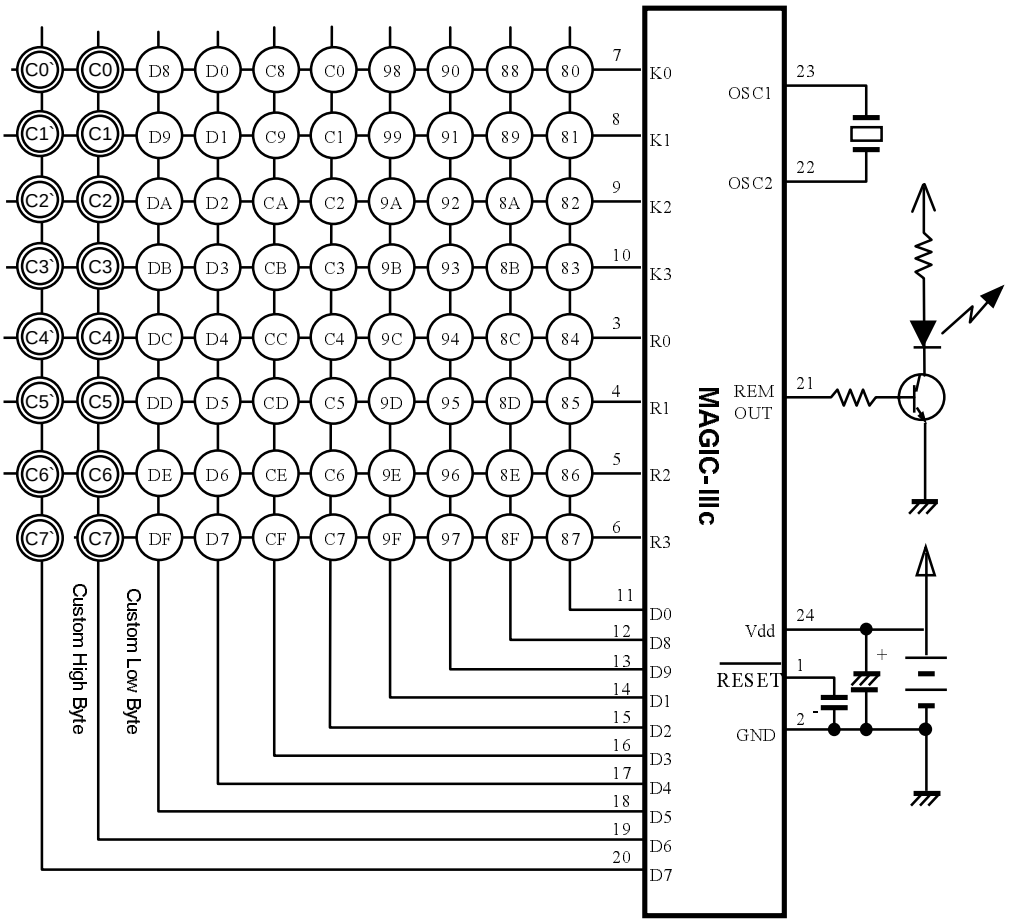
Tf = 108ms @455KHz

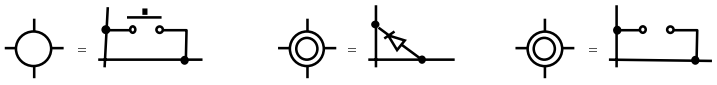
3. Pin Description

Pin No.	Pin	Funcrion	Remark
1	$\overline{\text{RESET}}$	Reset by input of ˝L˝	
2	GND	Reference voltage for all inputs & outputs 0V	
3	R0	Key Input No. 4 pin	
4	R1	Key Input No. 5 pin	
5	R2	Key Input No. 6 pin	
6	R3	Key Input No. 7 pin	
7	K0	Key Input No. 0 pin	
8	K1	Key Input No. 1 pin	
9	K2	Key Input No. 2 pin	
10	K3	Key Input No. 3 pin	
11	D0	Key Scan Signal No. 0 pin	
12	D8	Key Scan Signal No. 1 pin	
13	D9	Key Scan Signal No. 2 pin	
14	D1	Key Scan Signal No. 3 pin	
15	D2	Key Scan Signal No. 4 pin	
16	D3	Key Scan Signal No. 5 pin	
17	D4	Key Scan Signal No. 6 pin	
18	D5	Key Scan Signal No. 7 pin	
19	D6	Key Scan Signal No. 8 pin	
20	D7	Key Scan Signal No. 9 pin	
21	REMOUT	Out of Transmission pin	
22	OSC2	Connect Resonator between OSC1 & OSC2	
23	OSC1	Connect Resonator between OSC1 & OSC2	
24	V_{DD}	Connect 3V power source	

4. Circuit Diagram & Key Data

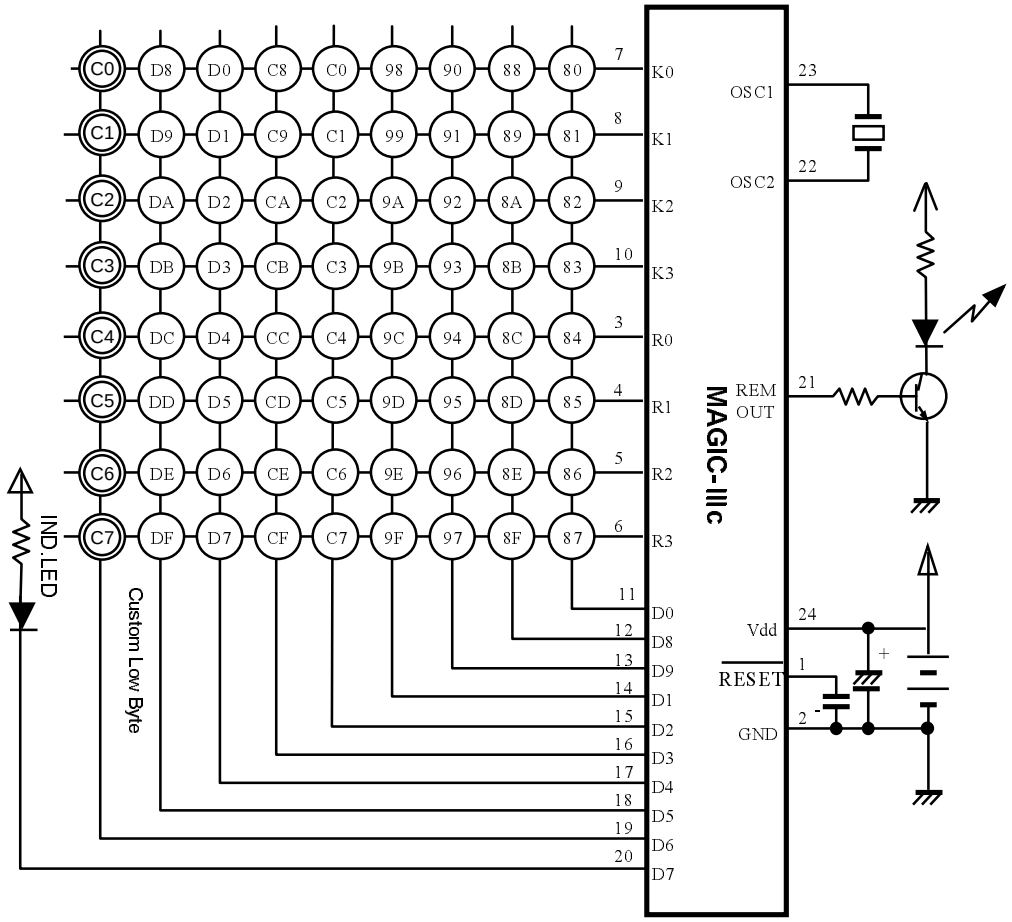
1) NEC 16bit for uPD6122-002

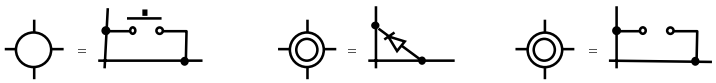




″High″ ″Low″

2) NEC 8bit for uPD6122-002





″High″ ″Low″

GMS300 MASK OPTION LIST

Code Name :

1. Device & Package

GMS30004 : 16SOP(150mil) ☐ 16SOP(300mil) ☐ 16DIP ☐
GMS30012 : 20SOP ☐ 20DIP ☐
GMS30112 : 20SOP ☐ 20DIP ☐
GMS30120 : 24SOP ☐ 24DIP ☐
GMS30140 : 24SOP ☐ 24DIP ☐

Please enter check marks as ✓

2. Inclusion of Pull up Resister

Port	R3	R2	R1	R0
Y/N				

Y : Yes
N : No

3. Release of Stop mode

Port	R3	R2	R1	R0	K3	K2	K1	K0
Y/N								

Y : Yes
N : No

4. Status of D port while Stop mode

Port	D7*	D6*	D5	D4	D3	D2	D1	D0
a/b								

a : State of ˝L˝ forcibly, b : Remain the state just before stop instruction
** : Market port is not available for GMS30004/012/112*

5. Reset of WDT due to REMOUT output

Y/N	

Y : Yes, N : No

6. Inclusion of condensor for Oscillation

Y/N	

Y : Yes, N : No

Data : 199 . .

Company Name :

Section Name :

Signature :