九齊科技股份有限公司
Nyquest Technology Co., Ltd.

USER MANUAL

NY4 Series

1-Ch Speech 4-bit MCU with 4~8 I/O

Version 1.8

Nov. 28, 2022

Revision History

Version#	Date	Description	Modified Page
1.0	2009/12/23	Formally release.	-
1.1	2010/2/12	1. NY4B provides large PWM current output. 2. NY4B provides large PWM current output. 3. NY4B provides large PWM current output. 4. Add the Noise-Filter description.	5 8 24 24
1.2	2010/7/26	1. Add new bodies of NY4B003A ~ NY4B011A. 2. Modify the initial value of $8 “BT” register from 4ms to “X”.	6 18
1.3	2010/9/08	1. Modify SRAM description. 2. Modify I/O port description for IR and reset options.	5 15
1.4	2013/11/29	1. Update NY4(B) Product List. 2. Modify the RPT descriptions of NY4 series.	6 18, 19
1.5	2014/11/21	Change IC body: NY4B018C, NY4B038C and NY4B058C.	6
1.6	2018/02/09	Update the DC Characteristics table.	8
1.7	2021/01/27	Remove NY4P085J, NY4B075B / 85B / 95B / 105B/ 115B / 125B / 145B / 165B.	-
1.8	2022/11/28	Add NY4P(C) only functions in Feature list.	5

Table of Contents

Chapter 1. Introduction

1.1 General Description

The NY4 series IC is a powerful 4-bit micro-controller based sound processor. There is only 1-channel speech with high quality direct-drive PWM output. By using the high fidelity ADPCM speech synthesis algorithm and a built-in noise filter, it can produce outstanding quality voices. Wide range sampling rate up to 44.1kHz is supported. The RISC MCU architecture is very easy to program and control, various applications can be easily implemented. There are 44 instructions, and most of them are executed in single cycle. Furthermore, a HALT mode (sleep mode) is designed to minimize power dissipation. Through accurate internal oscillation, external Rosc is unnecessary.

1.2 Features

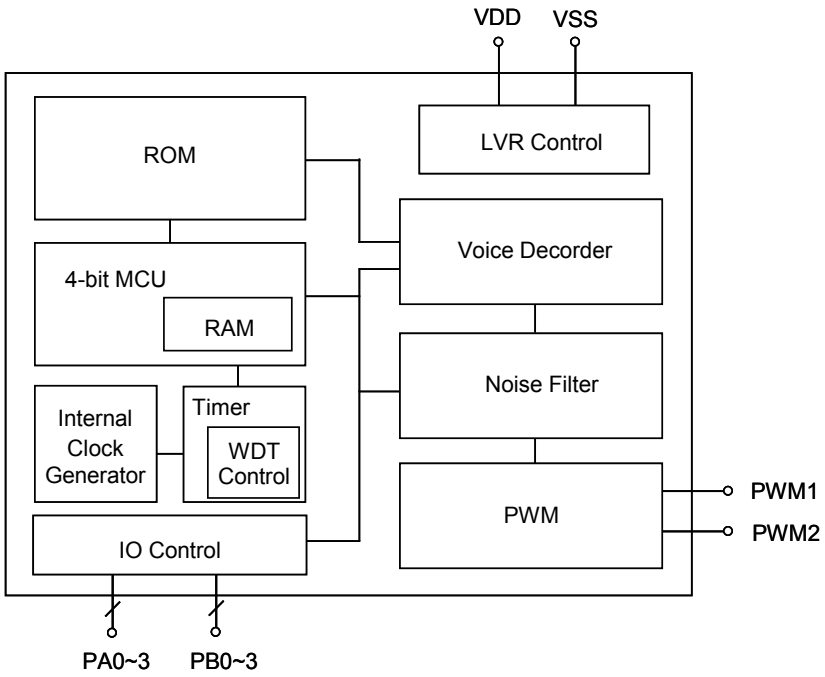
- Wide operating voltage range: 2.0V to 5.5V.
- 4-bit RISC type micro-controller with 44 instructions.
- 256Kx10-bit ROM maximum, program and voice data share the same ROM space.
- 96x4-bit RAM, divided into 2 pages.
- 1MHz instruction frequency.
- HALT mode to save power, less than 1uA@3V standby current.
- Precisely embedded oscillator with built-in resistor Rosc (+/- 1%).
- Low Voltage Reset, Watch-Dog Reset and I/O port reset are all supported to protect the system.
- Maximum 8 flexible I/Os with optional function: floating, pull-high, strong / weak pull-high, Reset input, IR carrier output. I/O's direction is controlled by registers. For the output port, users can select the large sink current output or normal drive current output.
- Infrared output: optional IR carrier frequency and optional data high/low IR output supported.
- 1-channel speech.
- New high fidelity ADPCM speech synthesis algorithm.
- Built-in the noise filter for less background noise at lower volume especially.
- One 9-bit hardware PWM output.
- Support large PWM current output. *(Only for NY4B series. NY4A don't provide this function.)*
- Mute mode speech algorithm to save ROM size.
- Quick-IO control supported.
- NY4P(C) supports LVD & Volume control. (Only for NY4P(C) series. Please reference NY4P(C) Datasheet for more detail)

1.3 Product List

IC Type	Time* (sec)	ROM (bits)	RAM (bits)	I/O	Synthesizer	PWM
NY4A003B	3.3	12K x 10	96 x 4	4	1-ch speech	1
NY4A005B	5	16K x 10	96 x 4	4	1-ch speech	1
NY4A008B	8.3	24K x 10	96 x 4	4	1-ch speech	1
NY4A011B	11.7	32K x 10	96 x 4	4	1-ch speech	1
NY4B003B	3.3	12K x 10	96 x 4	8	1-ch speech	1
NY4B005B	5	16K x 10	96 x 4	8	1-ch speech	1
NY4B008B	8.3	24K x 10	96 x 4	8	1-ch speech	1
NY4B011B	11.7	32K x 10	96 x 4	8	1-ch speech	1
NY4B018C	18.3	48K x 10	96 x 4	8	1-ch speech	1
NY4B025B	25	64K x 10	96 x 4	8	1-ch speech	1
NY4B038C	38.3	96K x 10	96 x 4	8	1-ch speech	1
NY4B045B	45	112K x 10	96 x 4	8	1-ch speech	1
NY4B058C	58.3	144K x 10	96 x 4	8	1-ch speech	1
NY4B065B	65	160K x 10	96 x 4	8	1-ch speech	1

* The voice duration is calculated at 6kHz by 4-bit ADPCM algorithm.

1.4 Block Diagram



1.5 Pad Description

Pin	ATTR.	Description
VDD#	Power	Positive power
GND#	Power	Negative power
PWM1	O	PWM1 output
PWM2	O	PWM2 output
PA0~3	I/O	Bit 0~3 for Port A
PB0~3	I/O	Bit 0~3 for Port B

* NY4A: PA0~PA3 (IR pad is shared with PA2, Reset pad is shared with PA3).

* NY4B: PA0~PB3 (IR pad is shared with PB2, Reset pad is shared with PB3).

1.6 Electrical Characteristics

The following lists the electrical characteristics of the NY4 EV chip. All the product's properties must refer to each part's datasheet.

1.6.1 Absolute Maximum Rating

Symbol	Parameter	Rated Value	Unit
V_{DD} - V_{SS}	Supply voltage	-0.5 ~ +6.0	V
Vin	Input voltage	V_{SS}–0.3V ~ V_{DD}+0.3	V
Top	Operating Temperature	0 ~ +70	°C
Tst	Storage Temperature	-25 ~ +85	°C

1.6.2 DC Characteristics

Symbol	Parameter		V_{DD}	Min.	Typ.	Max.	Unit	Condition
V_{DD}	Operating voltage			2.0	3	5.5	V	1 MHz
I_{sb}	Supply Current	Halt mode	3			1	uA	Sleep, no load
			4.5			1		
I_{op}		Operating mode	3		0.5		mA	1MHz, no loading
			4.5		1.5			
I_{il}	Input current (Internal pull-high)	Weak (750k ohms)	3		-4		uA	V_{il}=0v
			4.5		-11			
		Strong (33k ohms)	3		-90			
			4.5		-225			
I_{oh}	Output high current		3		-8		mA	V_{oh}=2.0V
			4.5		-12			V_{oh}=3.5V
I_{ol}	Output low current (Large current)		3		18		mA	V_{ol}=1.0V
			4.5		36			V_{ol}=1.0V
I_{PWM}	PWM output current (Normal)		3		60		mA	Load=8 ohms
			4.5		100			
I_{PWM}	PWM output current (Large)		3		70		mA	Load=8 ohms
			4.5		117			
△F/F	Frequency deviation by voltage drop (1MHz)		3		2		%	$\frac{Fosc(3.0v)-Fosc(2.4v)}{Fosc(3v)}$
			4.5		1			$\frac{Fosc(4.5v)-Fosc(3.0v)}{Fosc(4.5v)}$
△F/F	Frequency lot deviation (1MHz)		3	-1		1	%	$\frac{Fosc(3.0v)-1MHz}{1MHz}$
Fosc	Oscillation Frequency		-	0.90	1	1.05	MHz	V_{DD}=2.0~5.5V

Chapter 2. Hardware Architecture

2.1 Hardware Summary Table

Name	Function	Address
PC	Program counter	
VPR	Voice pointer register	
RPT	Multi-function register pointer	M[0x0~0x4]
RAM	96 nibbles RAM	
ROM	Program & data ROM	
SFR	Special function register	M[0x9]
PWM	PWM audio output	
INST	Instruction registers	
INST DEC	Instruction decoder	
PFLG	Play flag register	M[0x9]
Voice DEC	Voice decoder	
Clock Generator	Ring oscillator clock generator	
WDT	Watch-dog timer and reset generator	
BT	System base timer	M[0x8]
FD	Flash with Dynamic	M[0x8]
QIO Control	Quick-IO control code generator	
TM	Sample rate timer	
ROD1	ROM[7:4] data access register	M[0x6]
ROD2	ROM[9:8] data access register	M[0x7]
SYS Reset	System reset generator	
POR	Power reset generator	
LVDT	Low voltage detector and reset generator	
ACC	4-bit accumulator	
ALU	4-bit arithmetic logic unit	
C	Carry flag for arithmetic	
Z	Zero flag for arithmetic	
IR	Infrared transmit block	
I/O Ports	I/O port register	M[0xA~0xD]
VOL	Volume control register	M[0x5]
LVD	Low voltage detector and reset generator	M[0xE]

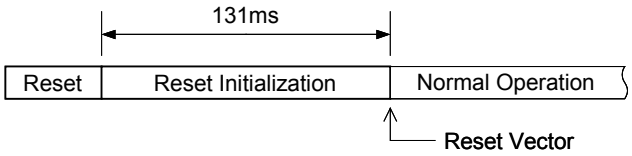
M[ ] : Memory register and the hex number 0x? between the brackets is its address.

Note：Only NY4P(C) series supports Volume Control and Low Voltage Detection (LVD) functions.

2.2 Clock Generator

The clock generator is a Ring oscillator, and users can only select the internal resistor (INT-R). The INT-R oscillator accuracy is up to ±1%.

2.3 System Reset



Reset Initialization Procedure

2.3.1 Power-On Reset (POR)

After Power-on, the power-on reset initialization will automatically be set out. After the system leaves the reset initialization procedure, it enters the normal operation and the program counter starts at the reset vector.

2.3.2 Low Voltage Reset (LVR)

When the system enters the normal operation, the power supply voltage must be kept in an effective working voltage range. When the power supply voltage is lower than the effective working voltage range, the system can't work properly. To prevent the system crash, we have a low voltage detector in the NY4 IC. When the detector detects a harmful low voltage supply, it will cause a low voltage reset. The so-called "low voltage" point of the NY4 IC is about 1.8v.

2.3.3 Watch-Dog Timer Reset (WDTR)

To recover from program malfunction, the NY4 IC supports an embedded watch-dog timer reset. The WDTR function always works with the program executing. Users have to clear the WDT periodically to prevent from timing up with a reset generation. Typically, the minimum time-up period of the WDT is about 25ms. Users can use WDT instruction to clear WDT.

It should be noted that WDT must be cleared carefully. Theoretically, it can't be placed at any loop other than main loop. One thing should be kept in mind, clear the WDT only when you make sure the execution of the MCU is still under the control of the program.

2.3.4 IO Port External Reset

The PA3 of NY4A and PB3 of the NY4B can be optioned as a reset pin. A reset pin should always be pulled-high in normal operation, whether to use the built-in internal pull-high resister option or to use the external one on PCB with the floating input option. When the reset pin falls to the ground level, it generates an external reset.

2.4 Address Pointer

The NY4 micro-controller contains a program counter (PC), and voice pointer (VPR). The length of each address pointer is 20-bit maximum, varies by the product parts. Not like the PC, the initial value of the VPR is unknown.

2.4.1 Program Counter (PC)

As a program instruction is executed, the PC will contain the address of the next program instruction to be executed. The PC starts from the reset vector (address 0x000000) after the system reset, and its value is increased by one every instruction cycle unless changed by a branch instruction.

Inst./Event	Function
JMP	Changes the LSB 14-bit of PC, and the reminder MSB bits keep their value.
CALL	The same functions as JMP. In addition, it pushes PC to RPT.
LDPC	Loads RPT to PC, so users can execute a long jump or jump by table.
RBPC	Reads back PC to RPT.

2.4.2 Voice Pointer Register (VPR)

Because NY4 is a 1-channel speech processor, a voice pointer is necessary for playing speech. When PLAY is executed, the system loads RPT to VPR. So users have to move the start address of the speech to RPT first. Besides, users can read VPR back by RBVP instruction.

2.5 Arithmetic Logic Unit (ALU)

The NY4 series provides a 4-bit arithmetic logic unit with a 4-bit accumulator to perform logic, unsigned arithmetic, data transfer and conditional branch operation. We have two flags (carry and zero) to indicate the result of the operation. One or two operands will be the data sources of the ALU operation. The operands can be ACC, RAM, register, or literal constant data.

2.5.1 ALU Instruction Summary

2.5.1.1 Logic Instruction

Instruction	Function	Flag Influenced
XORM m	A ← M[m] ⊕ A	Z
ANDM m	A ← M[m] & A	Z
ORM m	A ← M[m] | A	Z
XORL L	A ← A ⊕ L	Z
ANDL L	A ← A & L	Z
ORL L	A ← A | L	Z
ROLA	C ← A[3]; A ← {A[2:0], C}	C,Z
RORA	C ← A[0]; A ← {C, A[3:1]}	C,Z

2.5.1.2 Arithmetic Instruction

Instruction	Function	Flag Influenced
INCM m	A ← M[m] + 1	C, Z
DECM m	A ← M[m] - 1	C, Z
ADDM m	A ← A + M[m] + C	C, Z
ADDL L	A ← A + L + C	C, Z
INCA	A ← A + 1	C, Z
DECA	A ← A - 1	C, Z

2.5.1.3 Data Transfer Instruction

Instruction	Function	Flag Influenced
MVAM	M[m] ← A	
MVMA	A ← M[m]	Z
MVRM	M[m] ← R[r]	
MVMR	R[r] ← M[m]	
MVLR	R[r] ← L	
MVLA	A ← L	
RSTC	C ← 0	C
SETC	C ← 1	C

The width of the memory register address ‘r’ of MVRM, MVMR, and MVLR command is 2-bit and the MSB of the memory register is forced to be 0. So users can only use the three commands to handle RPT0~3. The width of the RAM or memory register address ‘m’ of MVRM, and MVMR command is 4-bit and the MSB 2-bit of the address is forced to be 0x3. Users can only use the two instructions to handle RAM or memory register of address 0x30~0x3F, but the RAM page is still working.

2.5.1.4 Conditional Branch Instruction

Instruction	Function	Flag Influenced
CPAM	Skip if A = M[m]	
CNAM	Skip if A != M[m]	
CPAL	Skip if A = L	
CNAL	Skip if A != L	
CPAB	Skip if (A & L) = 0	
CPCZ	Skip if C = 0	
CPZZ	Skip if Z = 0	
CNCZ	Skip if C != 0	C
CNZZ	Skip if Z != 0	C

A conditional branch instruction compares two data and skips next instruction if they are equal. The skip operation is making an instruction NOP, not jump across it.

⊕ : Exclusive OR bitwise logical operation

& : AND bitwise logical operation

| : OR bitwise logical operation

A : 4-bit Accumulator data

C : 1-bit carry flag data

L : 4-bit immediately literal data

M[m] : 4-bit RAM or memory register data at memory address m

R[r] : 4-bit memory register data at register address r

Z : 1-bit zero flag data

2.5.2 ALU Related Status Flag

Symbol	Flag	Description
C	Carry flag	C=1 if a carry-out occurs after an addition operation.
		C=0 if a borrow-in occurs after a subtraction operation.
Z	Zero flag	Z=1 if the result of an ALU operation is zero.

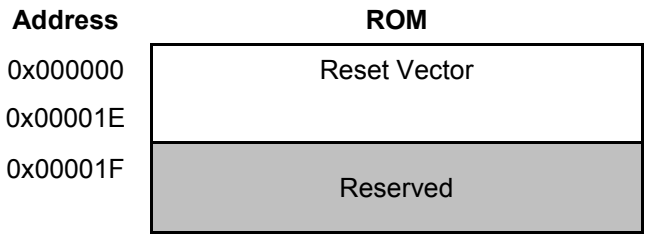
Besides RSTC and SETC commands directly assign the value of the carry flag, C is influenced by the arithmetic result. C means carry and also means the complement of borrow. If the addition operation larger than 0xF, C=1, and C=0 if the result $\leqq$15. If the subtraction operation smaller than 0, C=0, and C=1 if the result $\geqq$0.

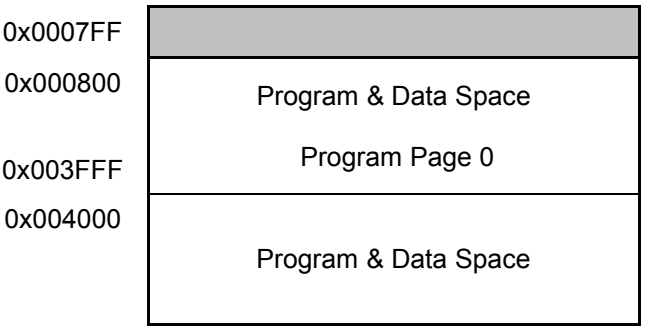
2.6 Memory Organization

There are maximum 256K words ROM, 96 nibbles of RAM and 14 nibbles of dedicated system control register. Besides, there are several registers without address allocation, and they can only be accessed by the special instructions. One of the registers is RAM page register (PG), and the other one is 8-bit sample rate timer (TM).

2.6.1 ROM

A large program/data/voice single ROM is provided and its structure is shown below. The reserved region contains system information and can't be utilized by users. The program page is limited by the unconditional branch instruction: JMP and CALL. Because it can only handle 14-bit length address of ROM, the program page size is 16K words.





2.6.2 RAM

Each page of RAM contains 48 nibbles. NY4 serial provides 96 nibbles of 2 pages. The page number (PG) register of RAM defined by the MPG instruction, and its initial value is 0. The address for RAM is 0x10~0x3F.

2.6.3 System Register

The system registers are located at the first 16 nibbles of the memory space, and the addresses of them are not affected by the memory page register (PG). By the memory mapping, user can perform any register addressing operations on them.

Address	Name	R/W	Description
0	RPT0	R/W	Register PoinTer [3:0]
1	RPT1	R/W	Register PoinTer [7:4]
2	RPT2	R/W	Register PoinTer [11:8]
3	RPT3	R/W	Register PoinTer [15:12]
4	RPT4	R/W	Register PoinTer [19:16]
5	VOL	R/W	4-level Volume control register
6	ROD1	R/W	ROM data bit [7:4] access register
7	ROD2	R/W	ROM data bit [9:8] access register
8	FD	R	Sign + 3 bits amplitude of the voice data (For Flash with Dynamic)
	BT	W	Base timing select register
9	SFR	R/W	BT, TOF, QIO, PLAY flag register
A	PA	R/W	PA data register
B	PAIO	R/W	PA I/O direction control register
C	PB	R/W	PB data register
D	PBIO	R/W	PB I/O direction control register
E	LVD	R/W	Low Voltage Detect control register

Note：Only NY4P(C) series supports Volume Control and Low Voltage Detection (LVD) functions.

2.7 IO Ports

There are 8 I/O ports at most, designated as PAx through PBx, and x=0~3. All the I/O ports can be configured as input or output by registers. For the input port, we provide an internal pull-high register option for convenience. For the output port, users can select the large sink current output or normal drive current output.

The PA2 or PB2 pin can be optioned as an infrared (IR) output pin. The PA3 or PB3 pin can be optioned as an external reset pin. A reset port can possess a pull-high resister or not, and an IR port can be large sink current or normal drive current output.

The pull-high resister of all the I/O ports has two kinds of option: weak and strong. The weak one is about 750kΩ @3V for normal application and the strong one is about 33kΩ @3V usually for key matrix function. When users configure the weak or strong pull-high resister, the pull-high resisters of all I/O ports are set as the option value.

For NY4A series

I/O	Option	Description
PA0, PA1	Normal I/O	-
	I/O port pull_high resister	Disable
		Enable
	Pull_high resister Weak/Strong	Weak
		Strong
PA2	IR output	choose 1 of 2
	Normal I/O	
	I/O port pull_high resister	Disable
		Enable
	Pull_high resister Weak/Strong	Weak
		Strong
PA3	Reset input	choose 1 of 2
	Normal I/O	
	I/O port pull_high resister	Disable
		Enable
	Pull_high resister Weak/Strong	Weak
		Strong

For NY4B series

I/O	Option	Description
PA0 ~ PA3 & PB0, PB1	Normal I/O	-
	I/O port pull_high resister	Disable
		Enable
	Pull_high resister Weak/Strong	Weak

I/O	Option	Description
		Strong
PB2	IR output	choose 1 of 2
	Normal I/O	
	I/O port pull_high resister	Disable
		Enable
	Pull_high resister Weak/Strong	Weak
		Strong
PB3	Reset input	choose 1 of 2
	Normal I/O	
	I/O port pull_high resister	Disable
		Enable
	Pull_high resister Weak/Strong	Weak
		Strong

2.8 Infrared Transmitter

The NY4 series provides an infrared transmit block which is used to send infrared signal. Users can option one of PA0~PB3 IO as an IR output. After that, users can option both the IR carrier frequency and IR Low/High carrier. The IR Low/High carrier means, for example. If users option the IR Low carrier, the IR output port sends infrared signal when the IO port register value is low, and vice versa.

I/O	Option	Description
IR	IR frequency	31.25 ~ 58.82 KHz
	IR low/high carrier	Low
		High

2.9 Volume Control

Only The NY4P(C) PWM supports 4 steps hardware volume control by the VOL register

Note：Only NY4P(C) series supports Volume Control and Low Voltage Detection (LVD) functions. Please verify LVD function on OTP chips.

2.10 LOW VOLTAGE DETECTOR (LVD)

There is one hardware voltage detector in NY4P(C) only. It offers seven levels for various application, 2.0v, 2.2v, 2.4v, 2.8v, 3.0v, 3.3v and 3.6v controlled by register $LVD. The voltage detection function has to be enabled first, then select specific level for application, the flag will go to high while VDD is lower than selected level. User can check power status by setting different level and monitoring the flag.

Note：Only NY4P(C) series supports Volume Control and Low Voltage Detection (LVD) functions. Please verify LVD function on OTP chips.

Chapter 3. System Control

3.1 Introduction

The PFLG, RPT and TM are audio control related registers. The Px and PxIO (x = A or B) are I/O ports registers. BTF, QIOF, TOF and BT are timing control related registers. The combination of RPT0~4 are multi-function register pointer. The C and Z are arithmetic associated flags. The PG is RAM access register. The ROD1 and ROD2 registers together with ACC are used to read the ROM data.

3.1.1 System Register Address Map

Addr	Name	R/W	Bit	Data	Description	Initial	Wake-up
0	RPT0	R/W	[3:0]	0/1	Register PoinTer[3:0]	X	U
1	RPT1	R/W	[3:0]	0/1	Register PoinTer[7:4]	X	U
2	RPT2	R/W	[3:0]	0/1	Register PoinTer[11:8]	X	U
3	RPT3	R/W	[3:0]	0/1	Register PoinTer[15:12]	X	U
4	RPT4	R/W	[3:0]	0/1	Register PoinTer[19:16]	X	U
5	VOL	R/W	[1:0]	0x0	Volume = 0%	0x02	U
				0x1	Volume = 50%		
				0x2	Volume = 100%		
				0x3	Volume = 200%		
6	ROD1	R/W	[3:0]	0/1	Register rOm Data[7:4]	X	U
7	ROD2	R/W	[1:0]	0/1	Register rOm Data[9:8]	X	U
8	FD	R	[3:0]	0/1	Sign + 3 bits amplitude of the voice data (For Flash with Dynamic)	X	U
	BT	W	[1:0]	0x0	BT = 0.125ms	X	U
				0x1	BT = 0.25ms		
				0x2	BT = 1ms		
				0x3	BT = 4ms		
9	SFR	R	[0]	0/1	TOF (Timer Overflow Flag)	0x0	U
			[1]	0/1	QIOF (QIO Flag)	0x0	U
			[2]	0/1	BTF (Base Timer overflow Flag)	0x0	U
			[3]	0/1	PFLG (Play FlaG)	0x0	U
		W	[3:0]	1	Write 1 to clear TO Flag	Clear	U
				1	Write 1 to clear QIO Flag	Clear	U
				1	Write 1 to clear BT Flag	Clear	U
				0/1	Resume / Pause voice playing	0	U
A	PA	R	[3:0]	0/1	PAIO = 1:Read port A input pad data	X	X
					PAIO = 0:Read port A output register	X	X
		W	[3:0]	0/1	Write to port A output register	X	U
B	PAIO	R/W	[3:0]	0/1	Port A direction = Output / Input	Input	U
C	PB	R	[3:0]	0/1	PBIO = 1:Read port B input pad data	X	X
					PBIO = 0:Read port B output register	X	X

Addr	Name	R/W	Bit	Data	Description	Initial	Wake-up
		W	[3:0]	0/1	Write to port B output register	X	U
D	PBIO	R/W	[3:0]	0/1	Write to port B output register	Input	U
E	LVD	R/W	[2:0]	0x0	Voltage Detect function disable	0x0	U
				0x1	Check Power < 2.0v => Flag = High		
				0x2	Check Power < 2.2v => Flag = High		
				0x3	Check Power < 2.4v => Flag = High		
				0x4	Check Power < 2.8v => Flag = High		
				0x5	Check Power < 3.0v => Flag = High		
				0x6	Check Power < 3.3v => Flag = High		
				0x7	Check Power < 3.6v => Flag = High		
			[3]	0/1	LVD Flag, disable to Low	0x0	U

Note：Only NY4P(C) series supports Volume Control and Low Voltage Detection (LVD) functions.

3.2 RPT

As implied in the name, RPT are multi-function registers. Users have to operate RPT in coordination with the instructions below.

The RPT of NY4A is 15-bit long, and the NY4B's RPT is 18-bit expect the NY4B125's, NY4B145's and NY4B165's are 19-bit. The redundant bits of RPT (RPT[19:15] of NY4A, RPT[19] of NY4B125, NY4B145 and NY4B165, RPT[19:18] of other NY4B parts) are un-writable and un-know if users read them. Whether the bits of RPT are redundant or useful, user have to initial all RPT(RPT[19:0]) to "0". Besides the instructions related to the TM only access bit [7:0] of the RPT, the others access all available bits. The RPT will be frequently accessed because of its multi-functionality, so the NY4 series provides 3 instructions to accelerate the access of RPT0~3: MVRM, MVMR and MVLR.

The CALL instruction pushes the PC to the RPT and jump to the subroutine address of the operand ‘a’. When the subroutine is finished, use LDPC to go back to the main program.

Inst./Event	Function
CALL	Pushes PC+2 to RPT.
LDPC	Loads RPT to PC.
RBPC	Reads back PC to RPT.
LDTM	Loads RPT[7:0] to TM.
PLAY	Loads RPT to VPR.
RBVP	Reads back VPR to RPT.
RBRO	Use RPT as address to read ROM data.

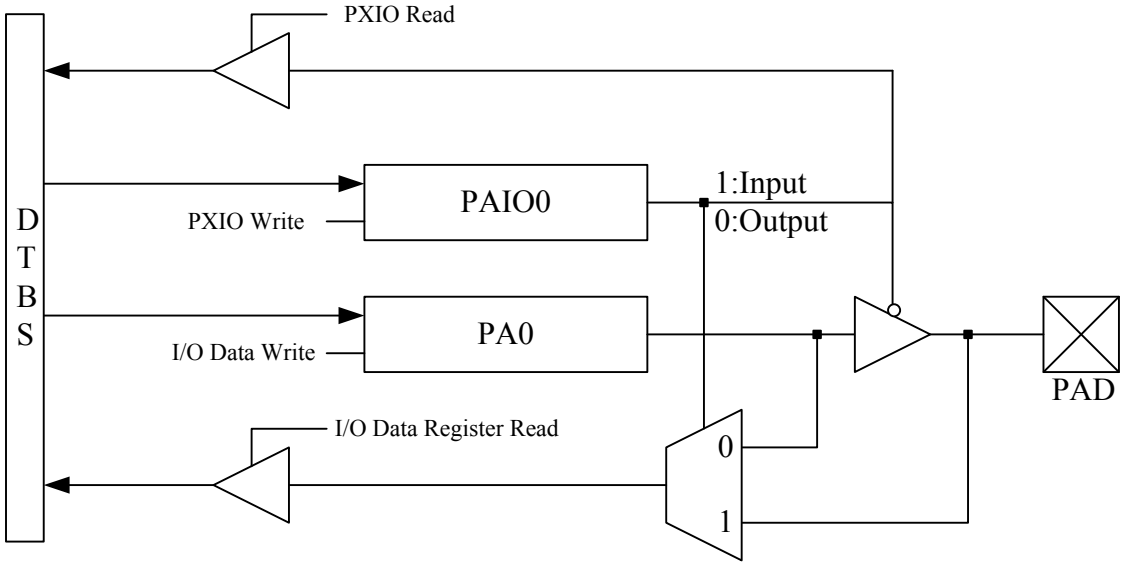
3.3 ROD

The NY4 series provides the RBRO instruction to read the ROM data out. When RBRO is executed, the system takes the RPT as ROM address, and the ROM data is loaded to ROD2, ROD1, and ACC. Bit[9:8] of the ROM data is loaded to ROD2, bit[7:4] to ROD1, and bit[3:0] to ACC. Using RBRO to read the data of the reserved ROM area out is unacceptable, and results in a reset. The RBRO instruction has a 1-bit operand 'n'. 0 means the RPT value keeps unchanged after RBRO, and 1 means the RPT adds 1.

3.4 I/O Ports Register

Before using an I/O port, configure its direction register first. The execution result of the read / write operation perform on the data register depends on the direction register. When an I/O port is configured as an input port (PxIO = 1), the data read from the data register is the pad status. Otherwise, it reads out the data stored in the data register.

The data register of an input port is only used for wake-up sequence. Because the difference between the pad and the register leads to a wake-up from the halt mode, users have to read the pad status and save back to the data register before entering the halt mode.



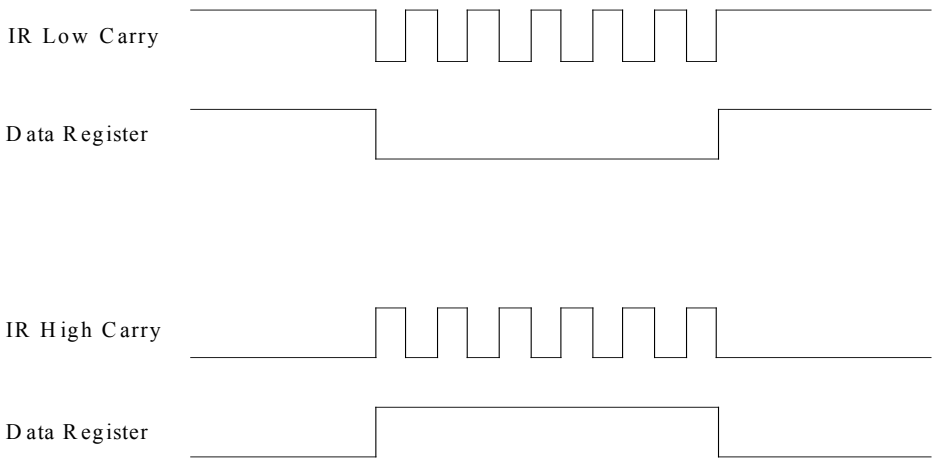
IO Port A0 Block Diagram

Contrastingly, the content of the data register is the pad status output by this output port.

If an IR carrier output is required, option an I/O port as an IR port first. An IR port is still an I/O port, which means the I/O direction still controlled by the direction register. Switching the direction register to output mode starts to output the IR signal.

One more Carry High / Low option provides the flexibility to connect the IR transmitter LED as Drive / Sink figure. If the IR LED is connected as sink manner, the Carry Low option is suggested to transmit the IR carrier

when the data register is Low. In this configuration, user can turn off the LED by programming a High to the data register to save power.



IR Carry Low / High Figure

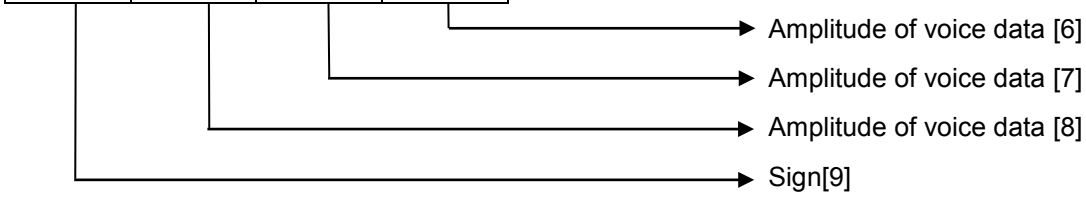
3.5 FD & BT

The reading source and the writing destination of the system register at address 0x8 are different

3.5.1 Flash with Dynamic (FD)

The Flash with Dynamic (FD) register can be read out for flash according to the level of voice data magnitude.

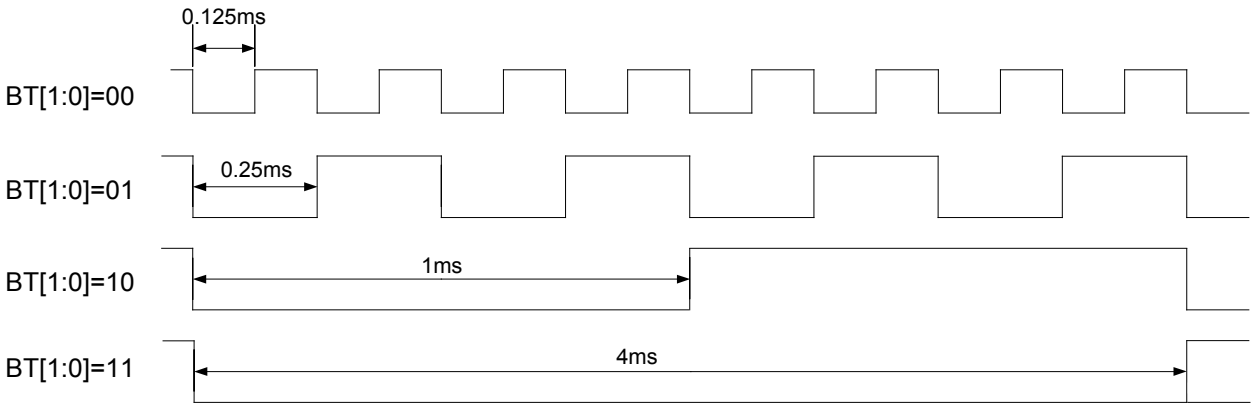
FD			
bit3	bit2	bit1	bit0
Read	Read	Read	Read



3.5.2 Base Timer (BT)

Write the 2-bit data of BT control register [1:0] to select BT overflow timing as 0.125ms, 0.25ms, 1ms or 4ms.

Note: The base timer must be initialized at program start. Users must write a certain value to configure BT register before reading the counter value of timer from BT register.



BT Timing Figure

3.6 Special Function Register (SFR)

The reading source and the writing destination of the system register of address 0x9 are different.

3.6.1 Special Function Register Related Flag

Reading from bit [0] shows the TOF value. The TOF represent the timer has been overflowed.

Reading from bit [1] shows the QIOF value. Quick-IO control code is a special code buried in the encoded speech data. When a QIO code is decoded in voice ROM, the QIOF arises.

Reading from bit [2] shows the BTF value. The BTF represent the base timer selected by the BTF source register has been overflowed.

Users can write 1 to clear TOF, QIOF, and BTF flag.

3.6.2 PFLG

Read value of bit [3] to check if speech is playing or not.

Write a “1” to PFLG to pause the speech, but keeps the last speech data unchanged.

Write a “0” to PFLG to resume the voice playing.

3.6.3 PG

There are 2 memory pages in NY4 series. The PG is not a system register or a memory register, so it can only be written by the MPG instruction and can’t be read. The 1-bit operand means the page users want to write to the PG register.

3.6.4 TM

The TM registers include a set 8-bit timer reload value latch and a set 8-bit counter. We load the latch and counter by LDTM. The TOF flag is set when the counter is overflow. Write a “1” to the TOF flag to clear the TOF flag, so it can be set again when the next time counter overflow. Set the TM as 0 to turn off the counter.

If the PFLG is set to 1 by the PLAY instruction, the TM register act as a sample rate timer also. The voice keeps playing until an END code is reached or be stopped by the STOP instruction. Loading a 0 to the TM to stop the timer counting after the PFLG is cleared to save power.

However, if a 0 is loaded into the TM when PFLG = 1, the TM stops to count and therefore the playing voice is paused. Loading any value other than zero resumes the playing voice.

LDTM loads the RPT[7:0] to the TM, and the TM can’t be read.

$TM = (F_{TCS} / F_{SR}) – 1$

TM : TM value in decimal

F_{TCS} : Frequency of the timer clock source

F_{SR} : Frequency of the sample rate

3.6.4.1 TM Table of Speech

TM	S.R. (KHz)
0xF9	4
0xA6	6
0x7C	8
0x63	10
0x52	12
0x46	14
0x3E	16
0x31	20

3.6.5 VOL

The VOL register specifies the digital volume control. The VOL has 4 steps. 0x0 means the mute mode and 0x3 is the loudest level. The relationship between the register value and the volume is as follows

Register Data	Volume
0x0	0%
0x1	50%
0x2	100%
0x3	200%

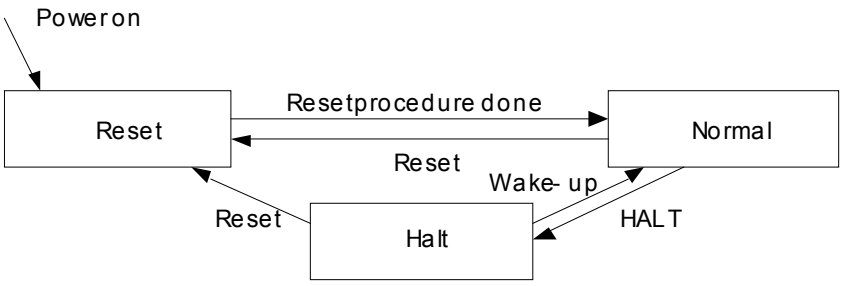
Note：Only NY4P(C) series supports Volume Control and Low Voltage Detection (LVD) functions.

3.6.6 LVD

The LVD register sets seven kinds of voltage level and provide a flag for reading to monitor the voltage level of VDD. The LVD[2:0] is to set voltage level. The LVD[3] is the flag which is shown out the VDD status. The flag will go to high if VDD is lower than selected voltage level.

Note：Only NY4P(C) series supports Volume Control and Low Voltage Detection (LVD) functions.

3.7 Power Saving Mode



Power Saving Mode Flow Chart

3.7.1 Halt Mode

The system enters the halt mode if the HALT command executed. The halt mode is also known as the sleep mode. As implied by the name, the IC falls asleep and the system clock is completely turned off, so all the IC functions are halted and it minimizes the power consumption.

The only way to wake-up the sleeping system is an input port wake-up. The IC keeps monitoring the input pads during the halt mode. If the input status of any input pad differs from the corresponding port register, the system will be waked-up. Then the succeeding instructions after the HALT instruction will be executed after the wake-up stable time (about 64us). So before executing the HALT instruction, users have to keep in mind to store the current input port statuses into port registers.

If the IC is waked-up from the halt mode by a reset port, it goes into the reset procedure.

Chapter 4. Audio Control

4.1 Speech Control

4.1.1 Speech Playing Procedure

Play a speech by setting the sample rate to TM, then executing the PLAY command. The playing will continue to play until the speech ends or users pause or stop it. When a speech ends, the PFLG falls to 0 automatically, so users can observe the PFLG to be aware of the voice play state. STOP automatically pulls the PFLG to 0 and stops the speech data to the middle value (0x100). Write a “1” to PFLG to pause the speech, but keeps the last speech data unchanged. Write a “0” to PFLG to resume the voice playing.

RBVP is a useful instruction to check the playing address of the speech data.

Step	Process	Instruction
1	Set TM in RPT1, RPT0	
2	Load TM from RPT	LDTM
3	Start to play	PLAY
4	Pause: Write 1 to PFLG	
5	Resume: Write 0 to PFLG	
6	Stop playing	STOP

The audio output is a 9-bit hardware PWM. For NY4B series, we provide an option of normal PWM current or large PWM current for different customer demand. The large PWM current consumes more current and makes sound louder. (NY4A don’t support the large PWM output.)

4.1.2 Speech Data

The NY4 series supports 9-bit PCM and encoded ADPCM speech data. Of course, the PCM speech has higher quality and occupies more ROM space than the ADPCM one. Use the encode software provided by the Nyquest to generate the PCM or ADPCM speech data. The voice start address is loading to VPR when executing the PLAY command.

There is a built-in Noise-Filter function. When users enable this option, hardware will suppress the noise to reduce the background noise automatically. Users can also disable this option up to the sound source.

4.2 Quick-IO Control

The NY4 series products support the quick-IO control. The QIO is a way to bury a control code into the speech data to handle the I/O ports. Use the Quick-IO software provided by Nyquest to utilize the function. As mentioned before, when playing a QIO buried data, the QIOF arises as a QIO code occurs. So users could do anything they want by managing the QIO control table.

4.3 Mute Mode

The NY4 series supports another special mute mode for speech. When a speech like the vocal or talk has a lot of suspension or silence, using the mute mode saves much ROM space. Turn on the mute mode option of the encode software to save your cost.

Chapter 5. Instruction Set

5.1 Instruction Classified Table

Item	Inst.	Op1	Op2	Operation	Inst. Length	Exec. Cycle	Oper. Flag	Flag Affected
Arithmetic Instructions								
1	INCM	6m		A = M +1	1	1		C, Z
2	DECM	6m		A = M - 1	1	1		C, Z
3	ADDM	6m		A = A + M + C	1	1	C	C, Z
4	XORM	6m		A = A ^ M	1	1		Z
5	ANDM	6m		A = A & M	1	1		Z
6	ORM	6m		A = A | M	1	1		Z
7	MVAM	6m		Move A to M	1	1		
8	MVMA	6m		Move M to A	1	1		Z
9	MVRM	4m	2r	Move R to M	1	1		
10	MVMR	4m	2r	Move M to R	1	1		
11	MVLR	4L	2r	Move L to R	1	1		
12	ADDL	4L		A = A + L + C	1	1	C	C, Z
13	XORL	4L		A = A ^ L	1	1		Z
14	ANDL	4L		A = A & L	1	1		Z
15	ORL	4L		A = A | L	1	1		Z
16	MVLA	4L		Move L to A	1	1		
17	RORA			C = A[0]; A = {C, A[3:1]}	1	1		C,Z
18	ROLA			C = A[3]; A = {A[2:0], C}	1	1		C,Z
19	RSTC			Reset C = 0	1	1		C
20	SETC			Set C = 1	1	1		C
21	INCA			A = A + 1	1	1		C, Z
22	DECA			A = A - 1	1	1		C, Z
Conditional Instructions								
23	CPAM	6m		Skip if A = M	1	1		
24	CNAM	6m		Skip if A ≠ M	1	1		
25	CPAL	4L		Skip if A = L	1	1		
26	CNAL	4L		Skip if A ≠ L	1	1		
27	CPAB	4L		Skip if (A & L) = 0	1	1		
28	CPCZ			Skip if C = 0	1	1	C	
29	CNCZ			Skip if C ≠ 0	1	1	C	
30	CPZZ			Skip if Z = 0	1	1	Z	
31	CNZZ			Skip if Z ≠ 0	1	1	Z	
Audio Instructions								
32	LDTM			Move RPT[7:0] to TM	1	1		
33	RBVP			Move VPR to RPT	2	3		
34	STOP			STOP playing CH immediately	1	1		
35	PLAY			Move RPT to VPR	1	3		

Item	Inst.	Op1	Op2	Operation	Inst. Length	Exec. Cycle	Oper. Flag	Flag Affected
Other Instructions								
36	MPG	1p		Set RAM page	1	1		
37	RBRO	1n		Move ROM[RPT] data to ROD and ACC	1	3		
38	JMP	14a		Jump to Address	2	2		
39	CALL	14a		Jump to Address, and Move PC to RPT	2	2		
40	LDPC			Move RPT to PC	1	2		
41	RBPC			Move PC to RPT	1	2		
42	HALT			Enter HALT mode	1	1		
43	CWDT			Clear WDT	1	1		
44	NOP			No operation	1	1		

A : 4-bit Accumulator data

C : 1-bit carry flag data

M : 4-bit RAM or memory register data

R : 4-bit memory register data

L : 4-bit immediately literal data

Z : 1-bit zero flag data

RPT : Multi-function register data

TM : 8-bit timer data of each channel

VPR : Voice address pointer of CH

ROM : 10-bit ROM data

ROD : ROM data access register data

PC : Program counter address pointer

a : ROM address

m : RAM or memory register address

n : data address Plus/Not plus 1

p : RAM page

r : Memory register address

5.2 Instruction Descriptions

5.2.1 Arithmetic Instructions

INCM m

Function: Add 1 to M of address m, and save the result back to A.

Operation: A ← M + 1

Operand: 0x0 ≤ m ≤ 0x3F

Words: 1

Cycles: 1

Operative Flags: None

Flags Affected: C, Z

Example: INCM m0

Before Instruction

M0=0x0

After Instruction

M0=0x1, C=0, Z=0

DECM m

Function: Subtract 1 from M of address m, and save the result back to A.

Operation: A ← M - 1

Operand: 0x0 ≤ m ≤ 0x3F

Words: 1

Cycles: 1

Operative Flags: None

Flags Affected: C, Z

Example: DECM m0

Before Instruction

M0=0x0

After Instruction

M0=0xF, C=0, Z=0

ADDM m

Function: Add A, C and M of address m, and save the result back to A.

Operation: A ← A + M + C

Operand: 0x0 ≤ m ≤ 0x3F

Words: 1

Cycles: 1

Operative Flags: C

Flags Affected: C, Z

Example: ADDM m0

Before Instruction

A=0x7, M0=0xA, C=0

After Instruction

A=0x7, M0=0x1, C=1, Z=0

XORM m

Function: Exclusive OR A with M of address m, and the result is save back to A.

Operation: A ← A ^ M

Operand: 0x0 ≤ m ≤ 3F

Words: 1

Cycles: 1

Operative Flags: None

Flags Affected: Z

Example: XORM m0

Before Instruction

A=0x3, M0=0xB

After Instruction

A=0x3, M0=0x8, Z=0

ANDM m

Function: AND A with M of address m, and save the result back to A.

Operation: A ← A & M

Operand: 0x0 ≤ m ≤ 0x3F

Words: 1

Cycles: 1

Operative Flags: None

Flags Affected: Z

Example: ANDM m0

Before Instruction

A=0x7, M0=0xA

After Instruction

A=0x7, M0=0x2, Z=0

ORM m

Function: Inclusive OR A with M of address m, and save the result back to A.

Operation: A ← A | M

Operand: 0x0 ≤ m ≤ 0x3F

Words: 1

Cycles: 1

Operative Flags: None

Flags Affected: Z

Example: ORM m0

Before Instruction

A=0x3, M0=0x8

After Instruction

A=0x3, M0=0xB, Z=0

MVAM m

Function: Move A to M of address m.

Operation: M ← A

Operand: 0x0 ≤ m ≤ 0x3F

Words: 1

Cycles: 1

Operative Flags: None

Flags Affected: None

Example: MVAM m0

Before Instruction

A=0x8

After Instruction

M0=0x8

MVMA m

Function: Move M of address m to A.

Operation: A ← M

Operand: 0x0 ≤ m ≤ 0x3F

Words: 1

Cycles: 1

Operative Flags: None

Flags Affected: Z

Example: MVMA m0

Before Instruction

M0=0x8

After Instruction

A=0x8

MVRM m, r

Function: Move R to M. Which R address MSB is 0 and M address MSB 2-bit is 0x3.

Operation: M ← R

Operand: 0x0 ≤ m ≤ 0xF

0x0r0x3

Words: 1

Cycles: 1

Operative Flags: None

Flags Affected: None

Example: MVRM m0, 0x0

Before Instruction

PG=0, RPT0=0x8

After Instruction

M30=0x8

MVMR m, r

Function: Move M to R. Which R address MSB is 0 and M address MSB 2-bit is 0x3.

Operation: R ← M

Operand: 0x0 ≤ m ≤ 0xF

0x0r0x3

Words: 1

Cycles: 1

Operative Flags: None

Flags Affected: None

Example: MVMR m1, 0x1

Before Instruction

PG=0, M31=0x8

After Instruction

RPT1=0x8

MVLR L, r

Function: Move the immediate constant value to R. Which R address MSB is 0.

Operation: R ← L

Operand: 0x0 ≤ L ≤ 0xF

0x0r0x3

Words: 1

Cycles: 1

Operative Flags: None

Flags Affected: None

Example: MVLR 0x7, 0x3

After Instruction

RPT3=0x7

ADDL L

Function: Add L and C to A.

Operation: A ← A + L+ C

Operand: 0x0 ≤ L ≤ 0xF

Words: 1

Cycles: 1

Operative Flags: C

Flags Affected: C, Z

Example: ADDL 0x9

Before Instruction

A=0xB, C=1

After Instruction

A=0x5, C=1, Z=0

XORL L

Function: Exclusive OR A with L.

Operation: A ← A ^ L

Operand: 0x0 ≤ L ≤ 0xF

Words: 1

Cycles: 1

Operative Flags: None

Flags Affected: Z

Example: XORL 0xA

Before Instruction

A=0x9

After Instruction

A=0x3, Z=0

ANDL L

Function: AND A with L.

Operation: A ← A & L

Operand: 0x0 ≤ L ≤ 0xF

Words: 1

Cycles: 1

Operative Flags: None

Flags Affected: Z

Example: ANDL 0xA

Before Instruction

A=0x0

After Instruction

A=0x0, Z=1

ORL L

Function: Inclusive OR AL.

Operation: A ← A | L

Operand: 0x0 ≤ L ≤ 0xF

Words: 1

Cycles: 1

Operative Flags: None

Flags Affected: Z

Example: ORL 0xA

Before Instruction

A=0x9

After Instruction

A=0xB, Z=0

MVLA L

Function: Move L to A.

Operation: A ← L

Operand: 0x0 ≤ L ≤ 0xF

Words: 1

Cycles: 1

Operative Flags: None

Flags Affected: None

Example: MVLA 0x7

After Instruction

A=0x7

RORA

Function: Right Rotate A to C and A.

Operation: C ← A[0]; A ← {C, A[3:1]}

Operand: None

Words: 1

Cycles: 1

Operative Flags: None

Flags Affected: C

Example: RORA

Before Instruction

A=0x5, C=0

After Instruction

A=0x2, C=1

RSTC

Function: Clear C to 0.

Operation: C ← 0

Operand: None

Words: 1

Cycles: 1

Operative Flags: None

Flags Affected: C

Example: RSTC

Before Instruction

C=1

After Instruction

C=0

ROLA

Function: Left Rotate A to C and A.

Operation: C ← A[3]; A ← {A[2:0], C}

Operand: None

Words: 1

Cycles: 1

Operative Flags: None

Flags Affected: C

Example: ROLA

Before Instruction

A=0x5, C=1

After Instruction

A=0xB, C=0

SETC

Function: Set C to 1.

Operation: C ← 1

Operand: None

Words: 1

Cycles: 1

Operative Flags: None

Flags Affected: C

Example: SETC

Before Instruction

C=0

After Instruction

C=1

INCA

Function: Add 1 to A.

Operation: A ← A + 1

Operand: None

Words: 1

Cycles: 1

Operative Flags: None

Flags Affected: C, Z

Example: INCA

Before Instruction

A=0xF

After Instruction

A=0x0, C=1, Z=1

DECA

Function: Subtract 1 from A.

Operation: A ← A - 1

Operand: None

Words: 1

Cycles: 1

Operative Flags: None

Flags Affected: C, Z

Example: DECA

Before Instruction

A=0x1

After Instruction

A=0x0, C=1, Z=1

5.2.2 Conditional Instructions

CPAM m

Function: Skip the next instruction if A equals M of address m.

Operation: Skip next if A=M

Operand: 0x0 ≤ m ≤ 0x3F

Words: 1

Cycles: 1, (2, 3)

Operative Flags: None

Flags Affected: None

Example: CPAM m0

Inst1

Inst2

After Instruction

If A≠M0, 'Inst1' is executed.

If A=M0, 'Inst1' is discarded, and 'Inst2' is executed.

CPAL L

Function: Skip the next instruction if A equals L.

Operation: Skip next if A=L

Operand: 0x0 ≤ L ≤ 0xF

Words: 1

Cycles: 1, (2, 3)

Operative Flags: None

Flags Affected: None

Example: CPAL 0x4

CALL a1

CALL a2

After Instruction

If A≠0x4 'CALL a1' is executed

If A=0x4 'CALL a1' is discarded, and 'CALL a2' is executed

CNAM m

Function: Skip the next instruction if A not equals M of address m.

Operation: Skip next if A≠M

Operand: 0x0 ≤ m ≤ 0x3F

Words: 1

Cycles: 1, (2, 3)

Operative Flags: None

Flags Affected: None

Example: CNAM m0

Inst1

Inst2

After Instruction

If A=M0, 'Inst1' is executed.

If A≠M0, 'Inst1' is discarded, and 'Inst2' is executed.

CNAL L

Function: Skip the next instruction if A not equals L.

Operation: Skip next if A≠L

Operand: 0x0 ≤ L ≤ 0xF

Words: 1

Cycles: 1, (2, 3)

Operative Flags: None

Flags Affected: None

Example: CNAL 0x4

CALL a1

CALL a2

After Instruction

If A=0x4 'CALL a1' is executed

If A≠0x4 'CALL a1' is discarded, and 'CALL a2' is executed

CPAB L

Function: Skip the next instruction if (A and L) not equals Zero.

Operation: Skip next if (A & L)≠0

Operand: 0x0≤L≤0xF

Words: 1

Cycles: 1, (2, 3)

Operative Flags: None

Flags Affected: None

Example: CPAB 0x4

CALL a1

CALL a2

After Instruction

If (A & L)≠0 'CALL a1' is executed

If (A & L) =0 'CALL a1' is discarded, and 'CALL a2' is executed

CNCZ

Function: Skip the next instruction if C not equals zero.

Operation: Skip next if C≠0

Operand: None

Words: 1

Cycles: 1, (2, 3)

Operative Flags: C

Flags Affected: None

Example: CNCZ

CALL a1

CALL a2

After Instruction

If C=0 'CALL a1' is executed

If C≠0 'CALL a1' is discarded, and 'CALL a2' is executed

CPCZ

Function: Skip the next instruction if C equals zero.

Operation: Skip next if C=0

Operand: None

Words: 1

Cycles: 1, (2, 3)

Operative Flags: C

Flags Affected: None

Example: CPCZ

CALL a1

CALL a2

After Instruction

If C≠0 'CALL a1' is executed

If C=0 'CALL a1' is discarded, and 'CALL a2' is executed

CPZZ

Function: Skip the next instruction if Z equals zero.

Operation: Skip next if Z=0

Operand: None

Words: 1

Cycles: 1, (2, 3)

Operative Flags: Z

Flags Affected: None

Example: CPZZ

CALL a1

CALL a2

After Instruction

If Z≠0 'CALL a1' is executed

If Z=0 'CALL a1' is discarded, and 'CALL a2' is executed

CNZZ

Function: Skip the next instruction if Z not equals zero.

Operation: Skip next if Z≠0

Operand: None

Words: 1

Cycles: 1, (2, 3)

Operative Flags: Z

Flags Affected: None

Example: CNZZ

CALL a1

CALL a2

After Instruction

If Z=0 ‘CALL a1’ is executed

If Z≠0 ‘CALL a1’ is discarded, and ‘CALL a2’ is executed

5.2.3 Audio Instructions

LDTM

Function: Load RPT[7:0] value to TM

Operation: TM ← RPT[7:0]

Operand: None

Words: 1

Cycles: 1

Operative Flags: None

Flags Affected: None

Example: MVLR 0x5, 0x1

MVLR 0x3, 0x0

LDTM

After Instruction

RPT0=0x3, RPT1=0x5, TM=0x53

STOP

Function: Stop audio playing immediately, and force the audio data to 0x100.

Operation: Stop playing

PFLG ← 0

Audio data ← 0x100

Operand: None

Words: 1

Cycles: 1

Operative Flags: None

Flags Affected: None

Example: STOP

Before Instruction

PFLG=1

After Instruction

PFLG=0,

Audio data = 0x100

RBVP

Function: Read address in VPR to RPT.

Operation: RPT ← VPR

Operand: None

Words: 1

Cycles: 3

Operative Flags: None

Flags Affected: None

Example: RBVP

Before Instruction

VPR=0xABCDE

After Instruction

RPT=0xABCDE

PLAY

Function: Play an audio in CH. The voice data address should be loaded into RPT first.

Operation: VPR ← RPT

PFLG ← 1

Operand: None

Words: 1

Cycles: 3

Operative Flags: None

Flags Affected: None

Example: PLAY

Before Instruction

RPT=0x12345

After Instruction

VPR=0x12345, PFLG=1

5.2.4 Other Instructions

MPG p

Function: Change the RAM page to page p.

Operation: RAM PAGE ← p

Operand: p[0,1]

Words: 1

Cycles: 1

Operative Flags: None

Flags Affected: None

Example: MPG 1

MVAM 0x12

Before Instruction

A=0xC

After Instruction

M62=0xC

JMP a

Function: Unconditionally jump by a direct address a.

Operation: PC ← a

Operand: 0x0 ≤ a ≤ 0x3FFF

Words: 2

Cycles: 2

Operative Flags: None

Flags Affected: None

Example: JMP a1

Before Instruction

PC=a0

After Instruction

PC=a1

Note: PC[19:14] will not be changed

RBRO n

Function: Read ROM data out to A and ROD using the RPT as address (data pointer).

Operation: A ← ROM data [3:0]

ROD1 ← ROM data [7:4]

ROD2 ← ROM data [9:8]

RPT ← RPT + n

Operand: 0n1

Words: 1

Cycles: 3

Operative Flags: None

Flags Affected: None

Example: RBRO 1

After Instruction

A=ROM[3:0] @ RPT

ROD1=ROM[7:4] @ RPT

ROD2=ROM[9:8] @ RPT

RPT=RPT+1

CALL a

Function: Call subroutine by a direct address a, and save next address to RPT.

Operation: RPT ← PC+2

PC ← a

Operand: 0x0 ≤ a ≤ 0x3FFF

Words: 2

Cycles: 2

Operative Flags: None

Flags Affected: None

Example: CALL a1

Before Instruction

PC=a0

After Instruction

PC=a1, RPT=a0+2

Note: PC[19:14] will not be changed.

LDPC

Function: Load RPT to PC. Unconditionally jump by the indirect address RPT. The address should be loaded into RPT first.

Operation: PC ← RPT

Operand: None

Words: 1

Cycles: 2

Operative Flags: None

Flags Affected: None

Example: LDPC

Before Instruction

RPT=0x54321

After Instruction

PC=0x54321

RBPC

Function: Read address in PC to RPT.

Operation: RPT ← PC+1

Operand: None

Words: 1

Cycles: 2

Operative Flags: None

Flags Affected: None

Example: RBPC

Before Instruction

PC=0x01234

After Instruction

RPT=0x01235

HALT

Function: Enter the halt (sleep) mode.

Operation: Stop system clock

Operand: None

Words: 1

Cycles: 2

Operative Flags: None

Flags Affected: None

Example: HALT

After Instruction

The system enters the halt mode and the system clock is halted.

CWDT

Function: Clear Watch-Dog Timer

Operation: Clear WDT

Operand: None

Words: 1

Cycles: 1

Operative Flags: None

Flags Affected: None

Example: CWDT

After Instruction

Recount WDT reset timer to 25us.

NOP

Function: No operation.

Operation: None

Operand: None

Words: 1

Cycles: 1

Operative Flags: None

Flags Affected: None

Example: NOP

After Instruction

No operation for 1 cycle.