July 20, 2009 Ver 0.3

4-BIT SINGLE CHIP MICROCOMPUTERS

ADAM26PXX

USER`S MANUAL

- ADAM26P16
- ADAM26P20
- ADAM26P20S
- ADAM26P20T
- ADAM26P23
- ADAM26P24

ABOV SEMICONDUCTOR

1. OVERVIEW

The ADAM26PXX is remote control transmitter which uses CMOS technology.
The ADAM26PXX is suitable for remote control of TV, VCR, FANS, Air-conditioners, Audio Equipments, Toys, Games etc. The ADAM26PXX is MTP version.

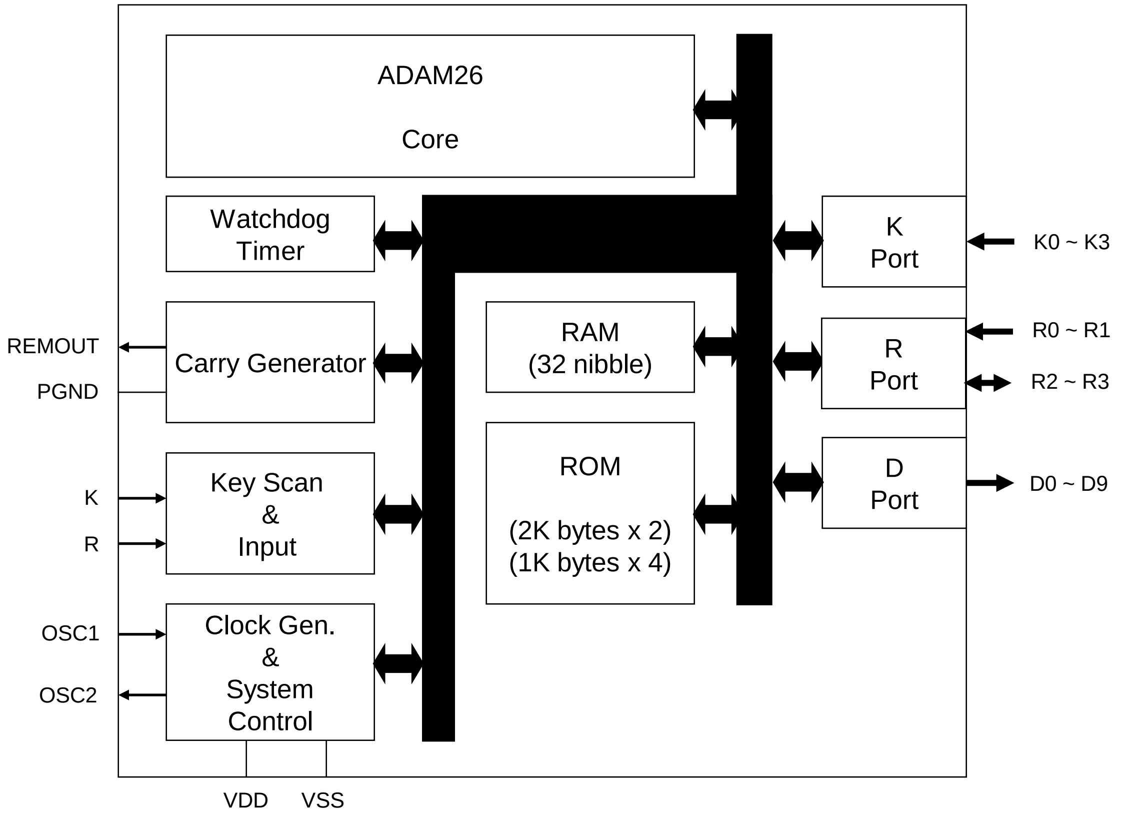
1.1. Features

- Program memory (MTP)
  - 2,048 bytes (2,048 x 8bit)
  - MTP (Multi Time programming) : 1K * 4, 2K * 2
- Data memory (RAM)
  - 32 nibble (32 x 4bit)
- 3 levels of subroutine nesting
- Operating frequency
  - 2.4MHz ~ 4MHz
- Instruction cycle
  - $f_{OSC}/48$
- Stop mode
- Released stop mode by key input
- Built in Power-on Reset circuit
- Built in Transistor for I.R LED Drive
  - I_{OL}=250mA at V_{DD}=3V and V_O=0.3V
- Built in Low Voltage reset circuit
- Built in a watch dog timer (WDT)
- Low operating voltage
  - 1.3 ~ 3.6V @ ADAM26PXX
- 16/20/24-SOP, 20-TSSOP Package.

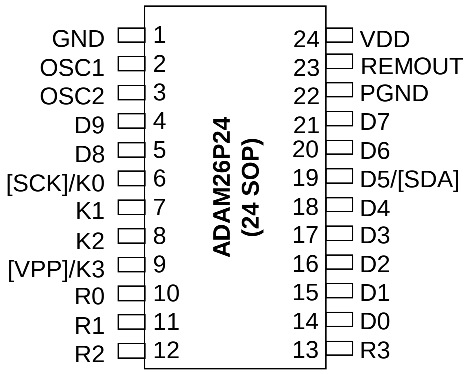
Series	ADAM26P24	ADAM26P23	ADAM26P20 ADAM26P20S ADAM26P20T	ADAM26P16
Program memory	2,048 x 8	2,048 x 8	2,048 x 8	2,048 x 8
Data memory	32 x 4	32 x 4	32 x 4	32 x 4
Input ports	7	7	7	5
I/O ports	2	2	2	2
Output ports	12	12	9	7
Package	24SOP(300mil)	24SOP(300mil)	20SOP(300mil) 20SOP (209mil) 20TSSOP(4.4mm)	16SOP(150mil)

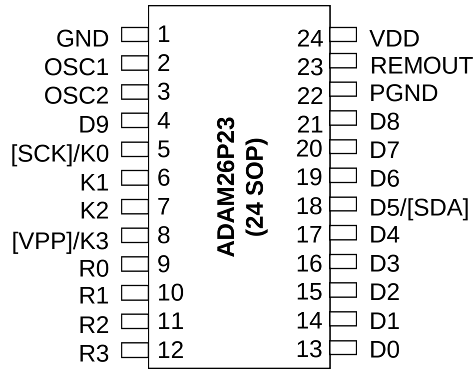
Table 1.1 ADAM26PXX series members

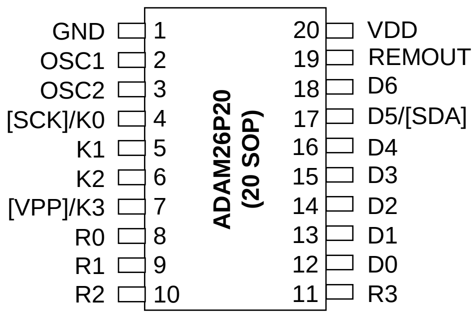
1.2. Block Diagram

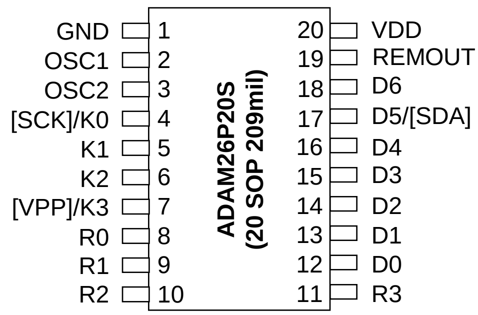


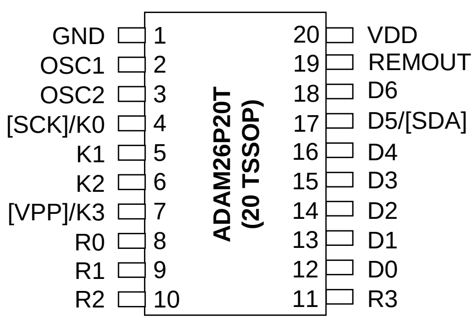
1.3. Pin Assignments (top view)

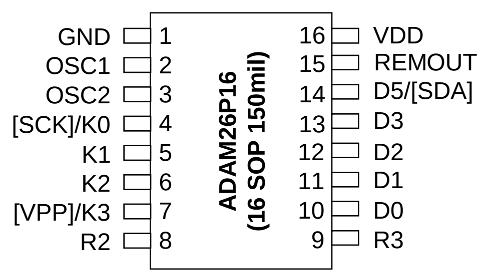




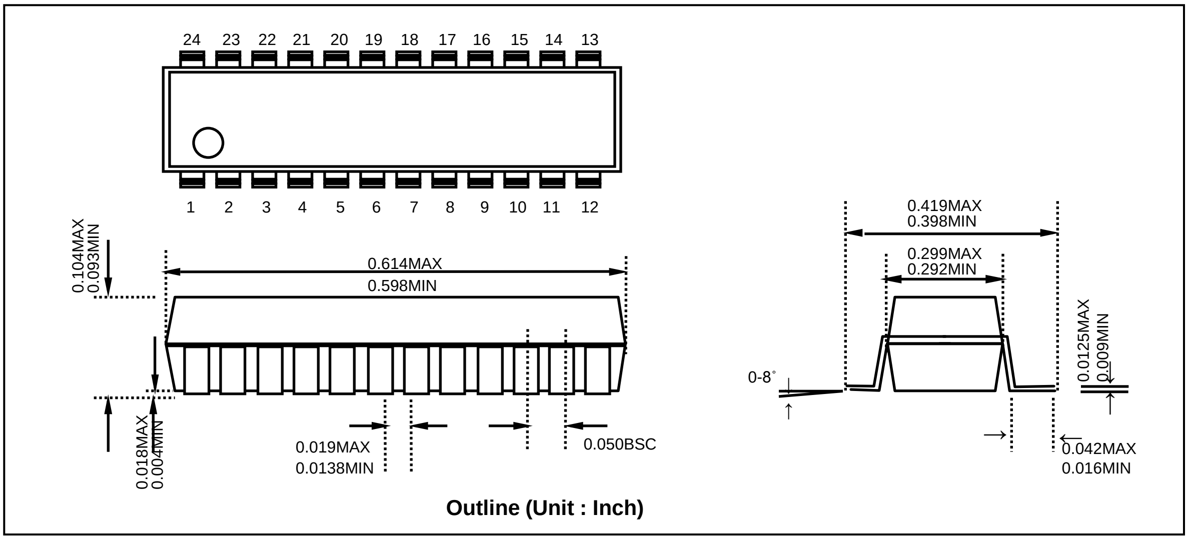




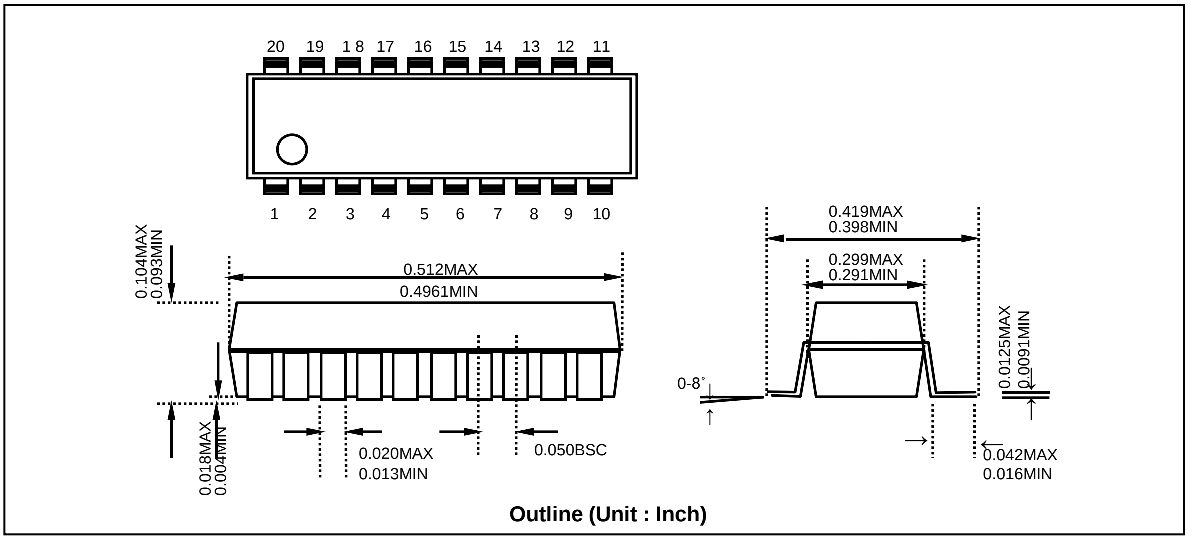




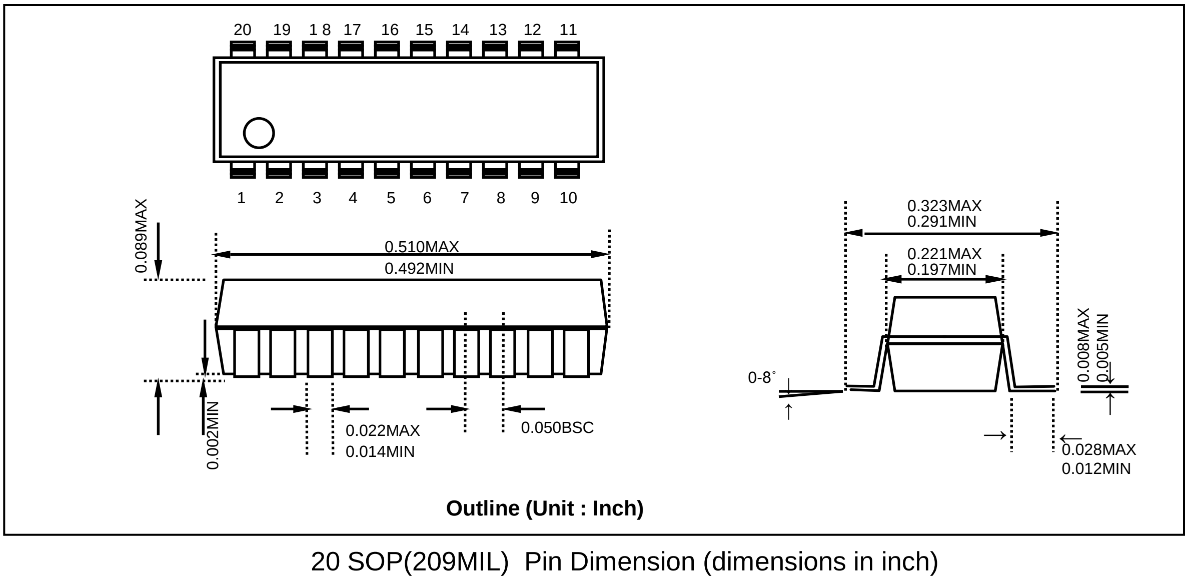
1.4. Package Dimension



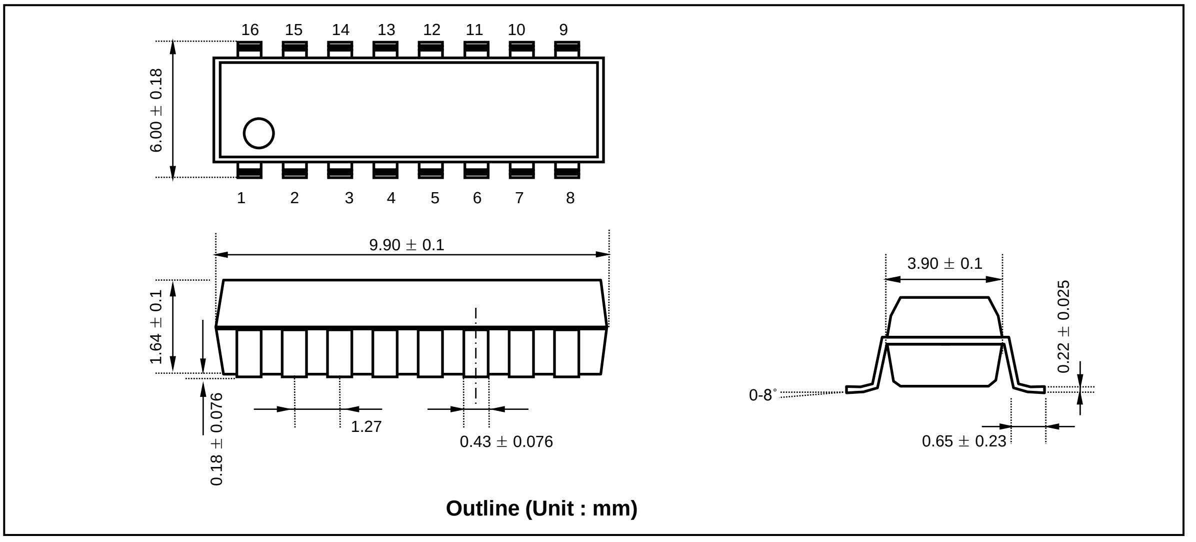
24 SOP(300MIL) Pin Dimension (dimensions in inch)



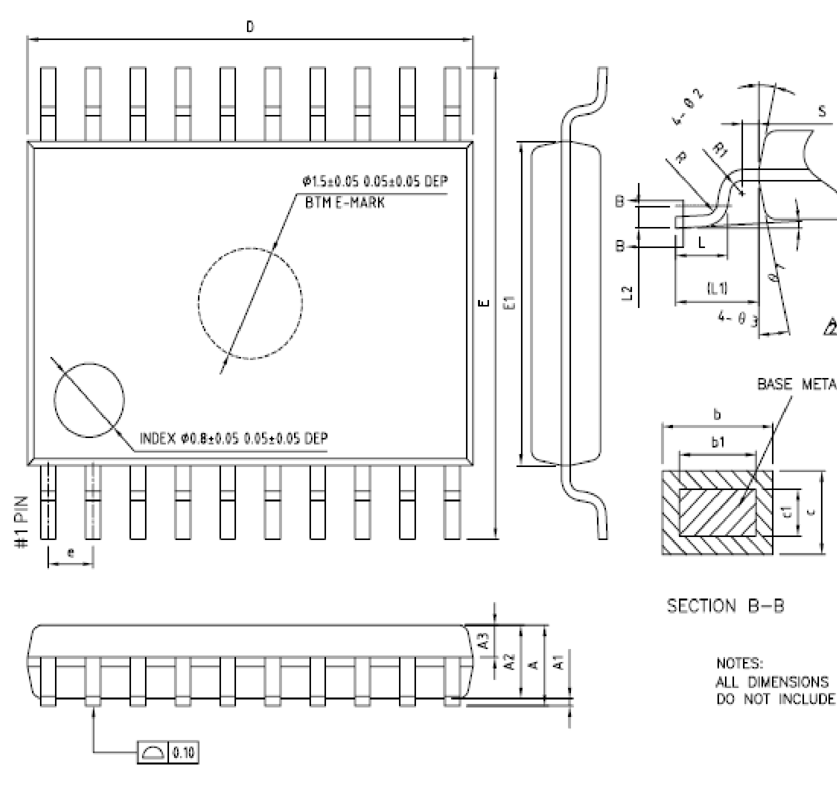
20 SOP(300MIL) Pin Dimension (dimensions in inch)



20 SOP(209MIL) Pin Dimension (dimensions in inch)



16 SOP(150MIL) Pin Dimension (dimensions in millimeters)



COMMON DIMENSIONS
(UNITS OF MEASURE=MILLIMETER)

SYMBOL	MIN	NOM	MAX
A	–	–	1.20
A1	0.05	–	0.15
A2	0.90	1.00	1.05
A3	0.34	0.44	0.54
b	0.20	–	0.28
b1	0.20	0.22	0.24
c	0.10	–	0.19
c1	0.10	0.13	0.15
D	6.40	6.50	6.60
E	6.20	6.40	6.60
E1	4.30	4.40	4.50
e	0.65BSC		
L	0.45	0.60	0.75
L1	1.00REF		
L2	0.25BSC		
R	0.09	–	–
R1	0.09	–	–
S	0.20	–	–
θ_1	0°	–	8°
θ_2	10°	12°	14°
θ_3	10°	12°	14°

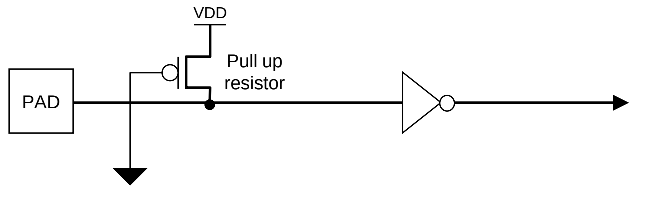
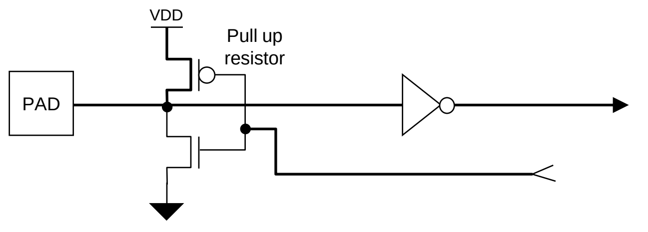
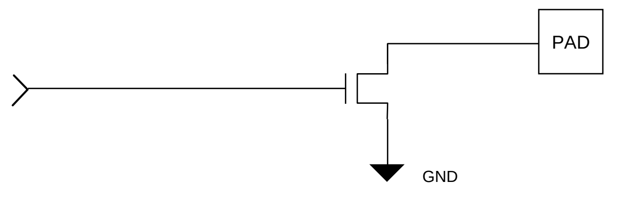
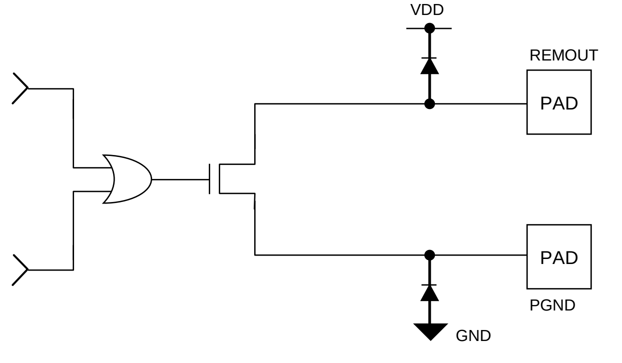
NOTES:
ALL DIMENSIONS REFER TO JEDEC STANDARD MO-153 AC
DO NOT INCLUDE MOLD FLASH OR PROTRUSIONS.

20 TSSOP(4.4mm) Pin Dimension (dimensions in millimeters)

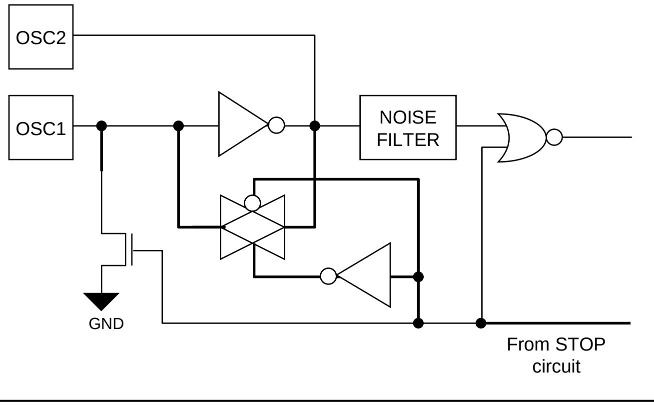
1.5. Pin Function

PIN NAME	INPUT OUTPUT	FUNCTION	@RESET	@STOP
K0~K3 R0~R1	Input	- 4-bit input Only Port. - CMOS input with pull-up resistor. - Each pin has STOP mode release function. (It is released by “L” input at STOP mode.)	Input (with Pull-up)	Input (with Pull-up)
R2~R3	I/O	-2-bit I/O Port. (Input mode is set only when each of them output “H”) - Each pin has STOP mode release function. - Output mode is set when each of them output “L” - When used as “output”, each pin can be set and reset independently	Input	-
D0 ~ D3	Output	- N-ch open drain output. - Each pin can be set and reset independently.	Low	Low
D4 ~ D9				Keep status before STOP
OSC1	Input	- Oscillator Input.	Oscillation	Low
OSC2	Output	- Oscillator Output.	Oscillation	High
REMOUT	Output	- High Current Pulse Output.	‘Hi-Z’ output	‘Hi-Z’ output
PGND	Power	-Ground pin for internal high current N-channel transistor. (connected to GND)	-	-
VDD	Power	- Positive power supply.	-	-
VSS	Power	- Ground	-	-

1.6. Pin Circuit

<table>
<tr><th>Pin Name</th><th>I/O</th><th>I/O circuit</th><th>Note</th></tr>
<tr><td>K
R0~R1</td><td>I</td><td>
</td><td>- Built in MOS Tr. for pull-up.</td></tr>
<tr><td>R2~R3</td><td>I/O</td><td>
</td><td>- CMODS output.
- “H” output at reset.
- Built in MOS Tr. for pull-up.</td></tr>
<tr><td>D0 ~ D9</td><td>O</td><td>
</td><td>- Open drain output.
- "L" output at reset.
- D0~D3 Ports are “L” output at Stop Mode.
- D4~D9 Ports keep the status before STOP at STOP Mode.</td></tr>
<tr><td>REMOUT
PGND</td><td>O</td><td>
</td><td>- Open drain output
- Output Tr. Disable at reset and Stop Mode.</td></tr>
<tr><td>OSC1

OSC2</td><td>I

O</td><td>
</td><td>- Built in feedback-resistor about 1㏁.</td></tr>
</table>

1.7. Electrical Characteristics

1.7.1. Absolute Maximum Ratings (Ta = 25℃)

Parameter	Symbol	Max. rating	Unit
Supply Voltage	V_{DD}	-0.3 ~ 5.0	V
Power dissipation	P_D	700 *	㎽
Input voltage	V_{IN}	-0.3 ~ V_{DD}+0.3	V
Output voltage	V_{OUT}	-0.3 ~ V_{DD}+0.3	V
Storage Temperature	T_{STG}	-65 ~ 150	℃

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

1.7.2. Recommended operating condition

Parameter	Symbol	Condition	Rating	Unit
Supply Voltage	V_{DD}	2.4MHz ~ 4MHz	1.3 ~ 3.6	V
Operating temperature	T_{opr}	-	-20 ~ +70	℃

1.7.3. DC Characteristics (Ta = 25℃, V_{DD}=3V)

Parameter		Symbol	Limits			Unit	Condition
			Min.	Typ.	Max.		
Input H current		I_{IH}	-	-	1	㎂	V_I=V_{DD}
Input Pull-up Resistance		R_{PU}	70	120	300	㏀	V_I=GND
OSC Feedback Resistance		R_{FD}	0.3	1.0	3.0	㏁	V_{OSC1}=GND, V_{OSC2}=V_{DD}
Input H voltage		V_{IH1}	2.1	-	-	V	-
Input L voltage		V_{IL1}	-	-	0.9	V	-
D output L voltage		V_{OL1}	-	0.15	0.4	V	I_{OL}=3㎃
OSC2 output L voltage		V_{OL2}	-	0.4	0.9	V	I_{OL}=150㎂
OSC2 output H voltage		V_{OH}	2.1	2.5	-	V	I_{OH}=-150㎂
REMOUT output L current		I_{OL}	-	250	-	㎃	V_{OL}=0.3V
REMOUT leakage current		I_{OLK1}	-	-	1	㎂	V_{OUT}=V_{DD}, Output off
D output leakage current		I_{OLK2}	-	-	1	㎂	V_{OUT}=V_{DD}, Output off
Current on STOP mode		I_{STP}	-	-	1	㎂	At STOP mode
Operating supply current		I_{DD}	-	0.7	1.5	㎃	f_{OSC} = 4MHz
System clock frequency	f_{OSC}/48	f_{OSC}	2.4	-	4	MHz	-

2. ARCHITECTURE

2.1. Program Memory

The ADAM26PXX can incorporate maximum 2,048 words (2 block × 16 pages × 64 words × 8bits) for program memory. Program counter PC (A0~A5) , page address register PA(A6~A9) and Block address register BA(A10) 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.

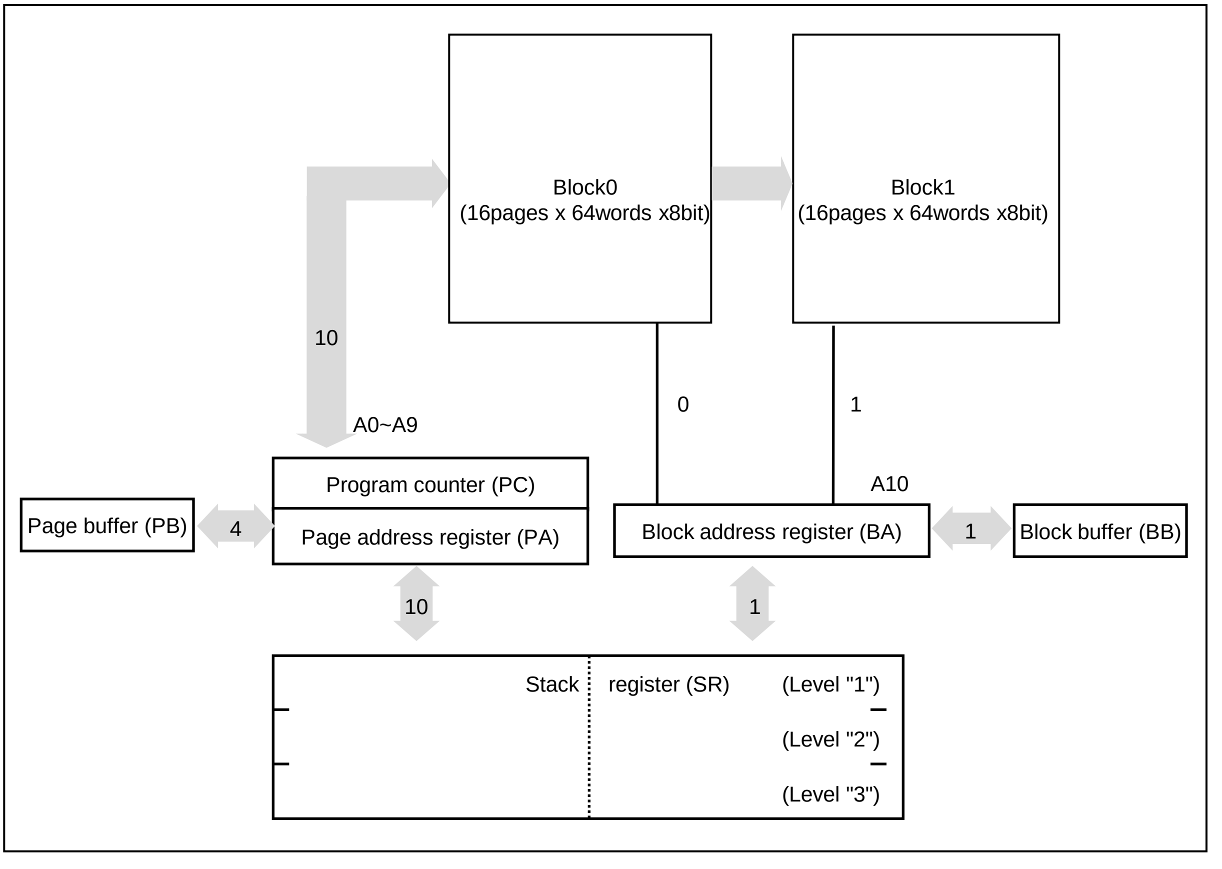


Fig 2-1 Configuration of Program Memory

2.2. Address Register

The following registers are used to address the ROM.

- Block address register (BA) :
  Holds ROM's Block number (0~1h) to be addressed.
- Block buffer register (BB) :
  Value of BB is loaded by an SEBB and REBB command when newly addressing a block.
  Then it is shifted into the BA when rightly executing a branch instruction (BR)
  and a subroutine call (CAL).
- 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.

2.2.1. Block address register and Block buffer register :

Address one of block #0 to #1 in the ROM by the 1-bit register.
Unlike the program counter, the block address register is not changed automatically.
To change the block address, take two steps such as
(1) writing in the block buffer what block to jump (execution of SEBB or REBB) and
(2) execution of BR or CAL, because instruction code is of eight bits so that block
can not 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.

2.2.2. 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 (execution of LPBI) and (2) execution of BR or CAL, because instruction code is of eight bits so that page and word can not 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.

2.2.3. 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 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.

2.2.4. Stack register :

This stack register provides three stages each for the program counter (6bits), the page address register (4bits) and block address (1bit) so that subroutine nesting can be made on three levels.

2.3. 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.

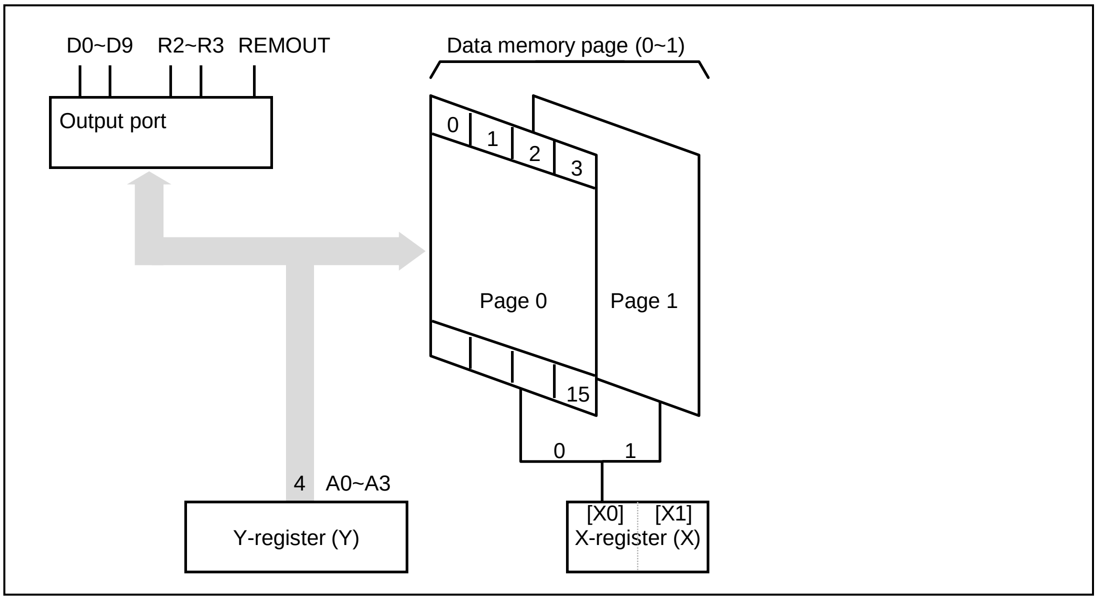


Fig 2-2 Composition of Data Memory

2.4. 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

Table2-1 Mapping table between X and Y register

2.5. Y-register (Y)

Y-register has 4 bits. It operates as a data pointer or a general-purpose register. Y-register specifies an 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.

2.6. Accumulator (A_{CC})

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

2.7. 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.)

2.7.1. 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)

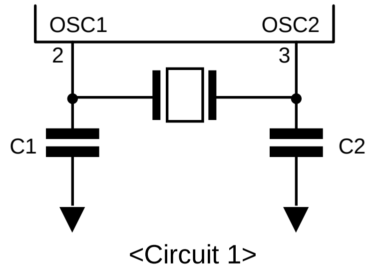
2.7.2. Status logic :

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

2.8. Clock Generator

The ADAM26PXX has an internal clock oscillator. The oscillator circuit is designed to operate with an external ceramic resonator.
Oscillator circuit is able to organize by connecting ceramic resonator to outside.

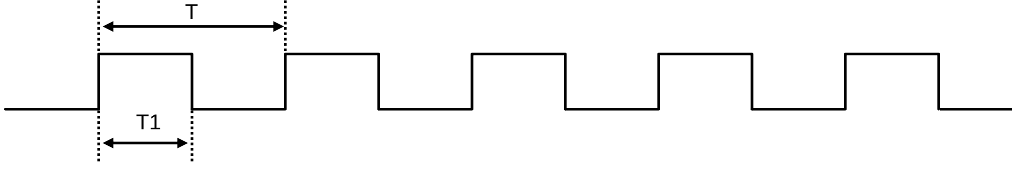
* It is necessary to connect capacitor to outside in order to change ceramic resonator, you must refer to a manufacturer`s resonator matching guide.



<Circuit 1>

2.9. 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	$T = 1/f_{PUL}$ = [$96/f_{OSC}$],	T1/T = 1/2
1	$T = 1/f_{PUL}$ = [$96/f_{OSC}$],	T1/T = 1/3
2	$T = 1/f_{PUL}$ = [$64/f_{OSC}$],	T1/T = 1/2
3	$T = 1/f_{PUL}$ = [$64/f_{OSC}$],	T1/T = 1/4
4	$T = 1/f_{PUL}$ = [$88/f_{OSC}$],	T1/T = 4/11
5	No Pulse (same to D0~D9)	
6	$T = 1/f_{PUL}$ = [$96/f_{OSC}$],	T1/T = 1/4
7	*Setting Prohibited*	

* Default value is "0"

Table 2-2 PMR selection table

2.10. Reset Operation

ADAM26PXX has two reset sources. One is a built-in Low VDD Detection circuit, another is the overflow of Watch Dog Timer (WDT).

All reset operations are internal in the ADAM26PXX.

2.11. Built-in Low VDD Reset Circuit

ADAM26PXX has a Low VDD detection circuit.

If VDD becomes Reset Voltage of Low VDD detection circuit in a active status, system reset occur and WDT is cleared.

When VDD is increased over Reset Voltage again, WDT is re-counted until WDT overflow, system reset is released.

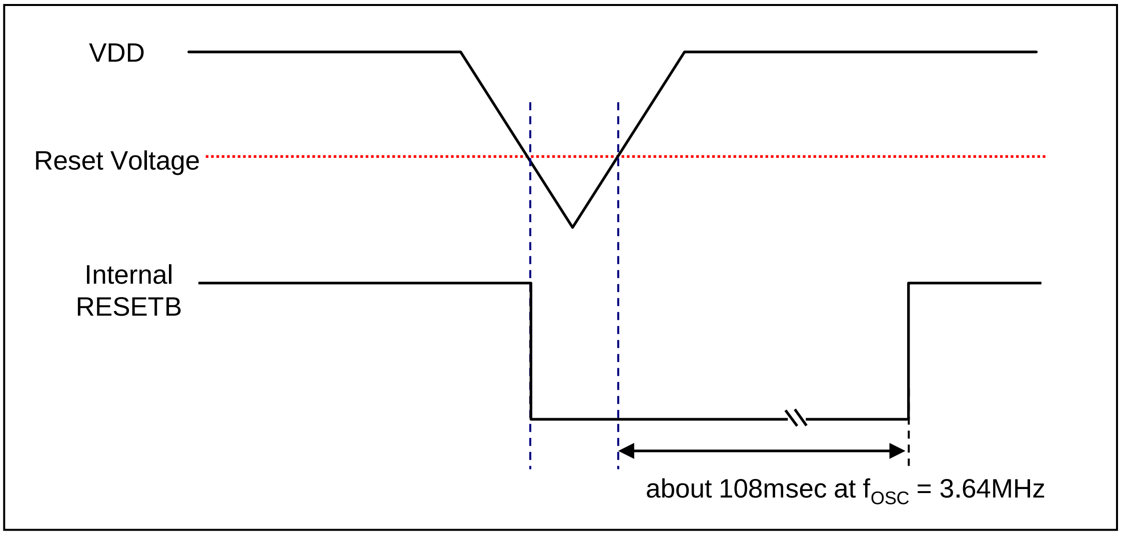


Fig 2-3 Low Voltage Detection Timing Chart.

2.12. Watch Dog Timer (WDT)

Watch dog timer is organized binary of 14 steps. The signal of $f_{OSC}/48$ cycle comes in the first step of WDT after WDT reset. If this counter was overflowed, reset signal automatically comes out so that internal circuit is initialized.

The overflow time is $8 \times 6 \times 2^{13}/f_{OSC}$ (108.026ms at f_{OSC} = 3.64MHz)

Normally, the binary counter must be reset before the overflow by using reset instruction (WDTR), Power-on reset pulse or Low VDD detection pulse.

* It is constantly reset in STOP mode. When STOP is released, counting is restarted. (Refer to 2.14. STOP Operation)

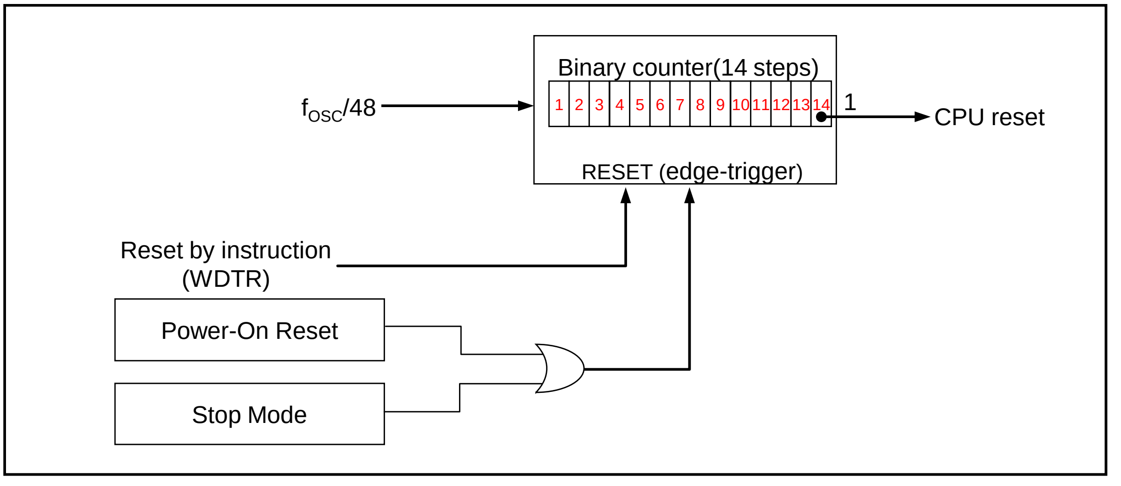


Fig 2-4 Block Diagram of Watch-dog Timer

2.13. STOP Operation

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 and REMOUT output is `High-Z` .
3. Part other than WDT and REMOUT output have a value before come into stop mode.
4. D0~D3 output are `Low` at STOP Mode.
5. D4~D9 output keep the status before STOP at STOP Mode.

`NOP` instruction should be follows STOP instruction for pre-charge time of Data Bus line.

ex) STOP : STOP instruction execution
NOP : NOP instruction

Stop mode is released when one of K or R input is going to `L`.

When stop mode released :

1. State of D0~D3 output and REMOUT output is return to state of before stop mode is achieved.
2. After $8\times6\times2^{10}/f_{OSC}$ time for stable oscillating, first instruction start to operate.
3. In return to normal operation, WDT is counted from zero.

When executing stop instruction, if any one of K,R input is `Low` state, stop instruction is same to NOP instruction.

2.14. Port Operation

Value of X - reg	Value of Y - reg	Operation
0 or 1	0 ~ 7	SO : D(Y) ← 1 (High-Z) RO : D(Y) ← 0
	8	REMOUT port repeats `H` and `L` in pulse frequency. (When PMR=5, it is fixed at `H` or `L`) SO : REMOUT(PMR) ← 0 RO : REMOUT(PMR) ← 1 (High-Z)
	9	SO : D0 ~ D9 ← 1 (High-Z) RO : D0 ~ D9 ← 0
	C ~ D	SO : R2(Y=C), R3(Y=D) ← 1 RO : R2(Y=C), R3(Y=D) ← 0
	E	SO : R2 ~ R3 ← 1 RO : R2 ~ R3 ← 0
	F	SO : D0 ~ D9 ← 1 (High-Z), R2 ~ R3 ← 1 RO : D0 ~ D9 ← 0, R2 ~ R3 ← 0
2 or 3	0	SO : D(8) ← 1 (High-Z) RO : D(8) ← 0
	1	SO : D(9) ← 1 (High-Z) RO : D(9) ← 0

3. INSTRUCTION

3.1. INSTRUCTION FORMAT

All of the 43 instruction in ADAM26PXX 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.

3.2. INSTRUCTION TABLE

The ADAM26PXX 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		SEBB	BB ←1	S
20		REBB	BB ← 0	S
21	Arithmetic	AM	A ← M(X,Y) + A	**C**
22		SM	A ← M(X,Y) - A	**B**
23		IM	A ← M(X,Y) + 1	**C**
24		DM	A ← M(X,Y) - 1	**B**
25		IA	A ← A + 1	S
26		IY	Y ← Y + 1	**C**
27		DA	A ← A - 1	**B**

	Category	Mnemonic	Function	ST[*1]
28	Arithmetic	DY	Y ← Y - 1	**B**
29		EORM	A ← A ⊕ M (X,Y)	S
30		NEGA	A ← $\overline{A}$ + 1	**Z**
31	Comparison	ALEM	TEST A ≤ M(X,Y)	**E**
32		ALEI i	TEST A ≤ i	**E**
33		MNEZ	TEST M(X,Y) ≠ 0	**N**
34		YNEA	TEST Y ≠ A	**N**
35		YNEI i	TEST Y ≠ i	**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 Refer to 2.14. Port Operation.

3.3. DETAILS OF INSTRUCTION SYSTEM

All 43 basic instructions of the ADAM26PXX 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

(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 ≤ i ≤ 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 ≤ i ≤ 15
Function : M(X,Y) ← i, Y ← 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 ≤ n ≤ 3
Function : X ← 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 ≤ n ≤ 3
Function : M(X,Y,n) ← 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 ≤ n ≤ 3
Function : M(X,Y,n) ← 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 ≤ n ≤ 3
Function : M(X,Y,n) ← 1?
ST ← 1 when M(X,Y,n)=1, ST ← 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 ← PB, PC ← a (Addr)
When ST = 0, PC ← PC + 1, ST ← 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 execute 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 : Ⅰ

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 ≤ i ≤ 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) SEBB

Naming : Set Block Buffer Register
Status : Set
Format : I
Function : BB ← 1
<Purpose> A new ROM page address is loaded into the block buffer register (BB).
This loading is necessary for a long branch or call instruction.
<Comment> The BB register is set to 1

(20) REBB

Naming : Reset Block Buffer Register
Status : Set
Format : I
Function : BB ← 0
<Purpose> A new ROM page address is loaded into the block buffer register (BB).
This loading is necessary for a long branch or call instruction.
<Comment> The BB register is set to 0

(21) AM

Naming : Add Accumulator to Memory and Status 1 on Carry
Status : Carry to status
Format : I
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.

(22) SM

Naming : Subtract Accumulator to Memory and Status 1 Not Borrow
Status : Carry to status
Format : I
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 complement 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″.

(23) IM

Naming : Increment Memory and Status 1 on Carry
Status : Carry to status
Format : I
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.

(24) DM

Naming : Decrement Memory and Status 1 on Not Borrow
Status : Carry to status
Format : I
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.

(25) IA

Naming : Increment Accumulator
Status : Set
Format : Ⅰ
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.

(26) IY

Naming : Increment Y-Register and Status 1 on Carry
Status : Carry to status
Format : Ⅰ
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.

(27) DA

Naming : Decrement Accumulator and Status 1 on Borrow
Status : Carry to status
Format : Ⅰ
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″.

(28) DY

Naming : Decrement Y-Register and Status 1 on Not Borrow
Status : Carry to status
Format : I
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.

(29) EORM

Naming : Exclusive or Memory and Accumulator
Status : Set
Format : I
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.

(30) NEGA

Naming : Negate Accumulator and Status 1 on Zero
Status : Carry to status
Format : I
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″.

(31) ALEM

Naming :	Accumulator Less Equal Memory	
Status :	Carry to status	
Format :	Ⅰ	
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 complement 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.	

(32) 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 complement 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.	

(33) MNEZ

Naming :	Memory Not Equal Zero	
Status :	Comparison results to status	
Format :	Ⅰ	
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 transferred to the status. Unless it is zero, the status is set.	

(34) YNEA

Naming : Y-Register Not Equal Accumulator
Status : Comparison results to status
Format : Ⅰ
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.

(35) 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.

(36) LAK

Naming : Load Accumulator from K
Status : Set
Format : Ⅰ
Function : A ← K

<Comment> Data on K are transferred to the accumulator

(37) LAR

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

(38) SO

Naming : Set Output Register Latch

Status : Set

Format : Ⅰ

Function :

D(Y) ← 1	0 ≤ Y ≤ 5
REMOUT ← 1(PMR=5)	Y = 8
D0~D4 ← 1 (High-Z)	Y = 9 or F
R(Y) ← 1	Ch ≤ Y ≤ Dh
R(Y) ← 1	Y = Eh
D0~D9,R2~R3 ← 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 or F.
When Y is between Ch and Dh, one of R2 and R3 is set to logic 1.
When Y is Eh, R2 and R3 is set to logic 1.
When Y is Fh, All D output and R2 and R3 is set to 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 or F, selects all D port.
Data in Y-register, when between Ch and Dh, selects an appropriate R port.
Data in Y-register, when it is Eh, selects all of R2~R3.
Data in Y-register, when it is Fh, selects all of D0~D9 and R2~R3.

(38) RO

Naming : Set Output Register Latch

Status : Set

Format : Ⅰ

Function :

D(Y) ← 0	0 ≤ Y ≤ 5
REMOUT ← 0(PMR=5)	Y = 8
D0~D4 ← 0	Y = 9 or F
R(Y) ← 0	Ch ≤ Y ≤ Dh
R(Y) ← 0	Y = Eh
D0~D9,R2~R3 ← 0	Y = Fh

<Purpose>
A single D output line is set to logic 0, if data of Y-register is between 0 to 7.
REMOUT port is set to logic 0, if data of Y-register is 8.
All D output line is set to logic 0, if data of Y-register is 9 or F.
When Y is between Ch and Dh, one of R2 and R3 is set to logic 0.
When Y is Eh, R2 and R3 is set to logic 0.
When Y is Fh, All D output and R2 and R3 is set to logic 0.

<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 or F, selects all D port.
Data in Y-register, when between Ch and Dh, selects an appropriate R port.
Data in Y-register, when it is Eh, selects all of R2~R3.
Data in Y-register, when it is Fh, selects all of D0~D9 and R2~R3.

(40) WDTR

Naming :	Watch Dog Timer Reset
Status :	Set
Format :	I
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 :	I
Function :	Operate the stop function
<Purpose>	Stopped oscillator, and little current.

(42) LPY

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

(43) NOP

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

3.4. Guideline for S/W

(1) All rams need to be initialized to any value in reset address for proper design.

(2) Make the output ports `High` after reset.

(3) Do not use WDTR instruction in subroutine.

(4) When you try to read input port changed from external condition, you must secure chattering time more than 200uS.

(5) To decrease current consumption, make the output port as high in normal routine except for key scan strobe and STOP mode.

(6) We recommend you do not use all 64 ROM bytes in a page.
It's recommend to add "BR $" at first and last address of each page.
Do not add "BR $" at reset address which is first address of "00" page of "0" bank.

(7) `NOP` instruction should be follows STOP instruction for pre-charge time of Data Bus line.

```
ex) STOP : STOP instruction execution
    NOP  : NOP  instruction
```