# 4 to 1 Analog Multiplexer Datasheet AMux4 V 1.50

| Resources | PSoC® Blocks | | | API Memory (Bytes) | | Pins (per External I/O) |
|---|---|---|---|---|---|---|
| | Digital | Analog CT | Analog SC | Flash | RAM | |
| CY8C29/27/24/22/21xxx, CY8C23x33, CY8CLED02/04/08/16, CY8CLED0xD, CY8CLED0xG, CY8CTST110, CY8CTMG110, CY8CTST120, CY8CTMG120, CY8CTMA120, CY8C21x45, CY8C22x45, CY8CPLC20, CY8CLED16P01, CY8C28x43, CY8C28x13, CY8C28x52, CY7C64215 | | | | | | |
| | 0 | 0 | 0 | 25 | 0 | 1 to 4 |
| CYWUSB6953 | 0 | 0 | 0 | 25 | 0 | 1 to 4 |
| CY8C26/25xxx | 0 | 0 | 0 | 25 | 0 | 1 to 4 |

For one or more fully configured, functional example projects that use this user module go to www.cypress.com/psocexampleprojects.

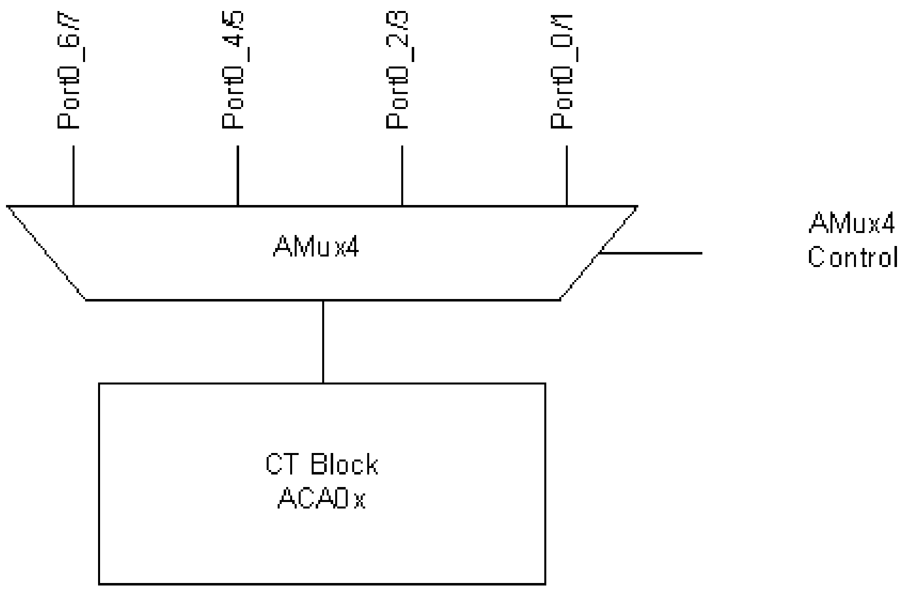
## Features and Overview

- High impedance input
- Input signals may be rail-to-rail
- May be used with RefMux to multiplex input signals to switch capacitor block
- Programmable control of input source

The AMux4 User Module provides a four input analog signal multiplexer to a Continuous Time (CT) block that can be controlled programmatically by way of an API. One of four input signals may be selected to the input of the amplifier in the CT block. These input signals are connected to fixed ports, depending on which column this user module is placed. This module can also be used in conjunction with a RefMux to route the multiplexed signals to the analog column bus.

The AMux4 User Module is only intended for use when the application must dynamically select from two or more ports during operation. If the input pin to the CT block is not changed during program operation, use the Device Editor to select the desired pin by left-clicking the AInMux_n mux.

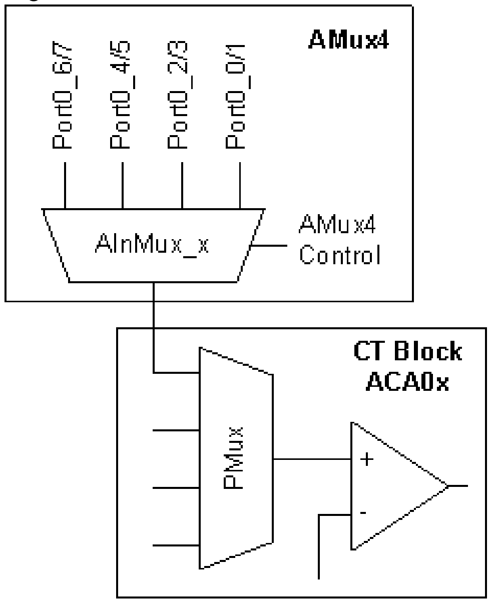
Figure 1. AMux4 Block Diagram



## Functional Description

The AMux4 User Module provides an API to control the analog input multiplexers (AInMux_x) for each analog column. The output of the AInMux_n is routed to the PMux (Positive input Mux) of the amplifier in the CT block. The PMux in the CT block can be configured to accept the input from the AMux4 User Module, by selecting the “AnalogInput_ColumnMux_n" for one of the inputs.

Figure 2. AMux4 Connection to CT Block



## DC and AC Electrical Characteristics

Unless otherwise specified in the following table, all limits guaranteed for $T_A$ = -40°C to +85°C, $V_{DD}$ = 5.0V ± 10%.

Table 1. 5.0V AMux4 AC and DC Electrical Characteristics

| Parameter | Conditions and Notes | Minimum | Typical | Limit | Units |
|---|---|---|---|---|---|
| Input Leakage | | -- | 0.1 | 5 | uAmps |
| Input Capacitance | | 0.5 | 1.7 | 8 | pF |
| Bandwidth | | -- | 10 | -- | MHz |
| Input Voltage Range | | 0 | -- | $V_{DD}$ | V |

Unless otherwise specified in the following table, all limits guaranteed for $T_A$ = -40°C to +85°C, $V_{DD}$ = 3.3V ± 10%.

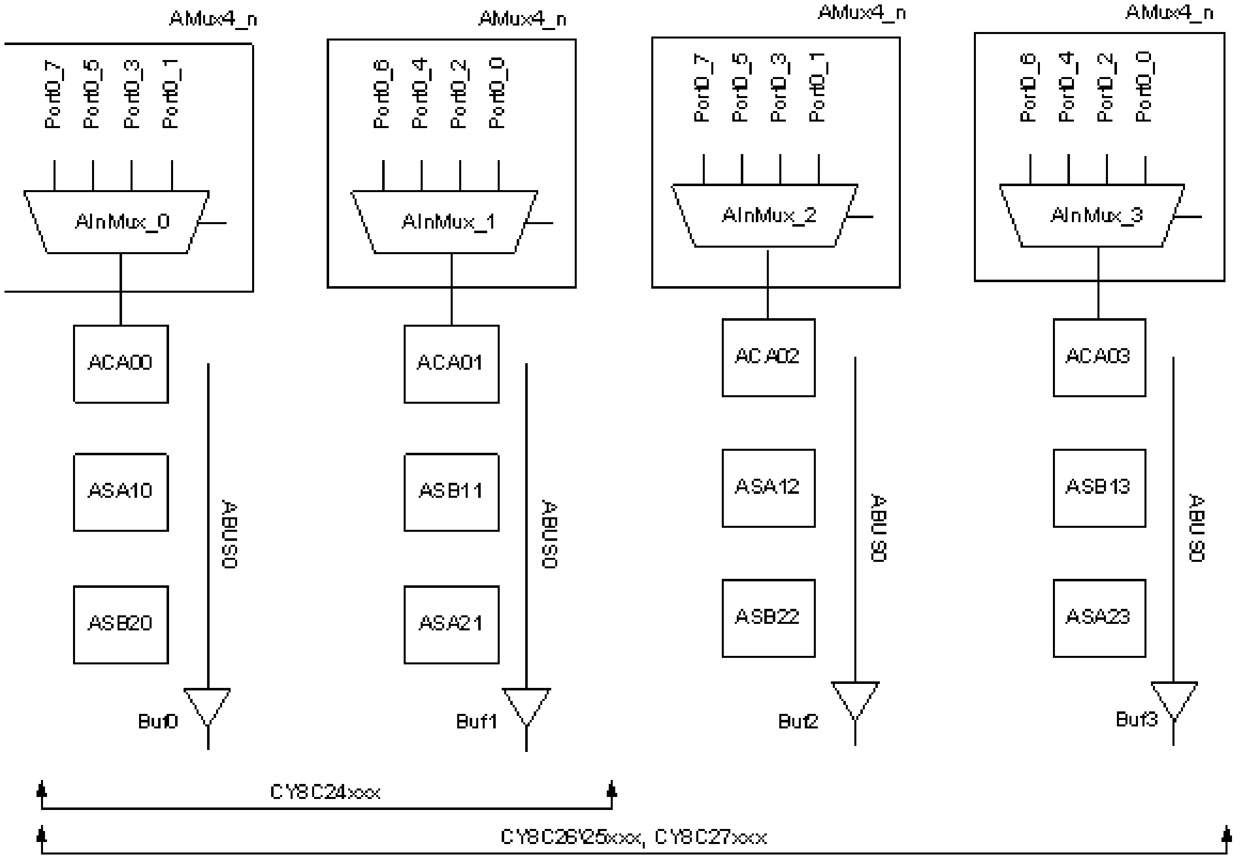
Table 2. 3.3V AMux4 AC and DC Electrical Characteristics

| Parameter | Conditions and Notes | Minimum | Typical | Limit | Units |
|---|---|---|---|---|---|
| Input Leakage | | -- | 0.1 | 5 | uAmps |
| Input Capacitance | | 0.5 | 1.7 | 8 | pF |
| Bandwidth | | -- | 10 | -- | MHz |
| Input Voltage Range | | 0 | -- | $V_{DD}$ | V |

## Placement

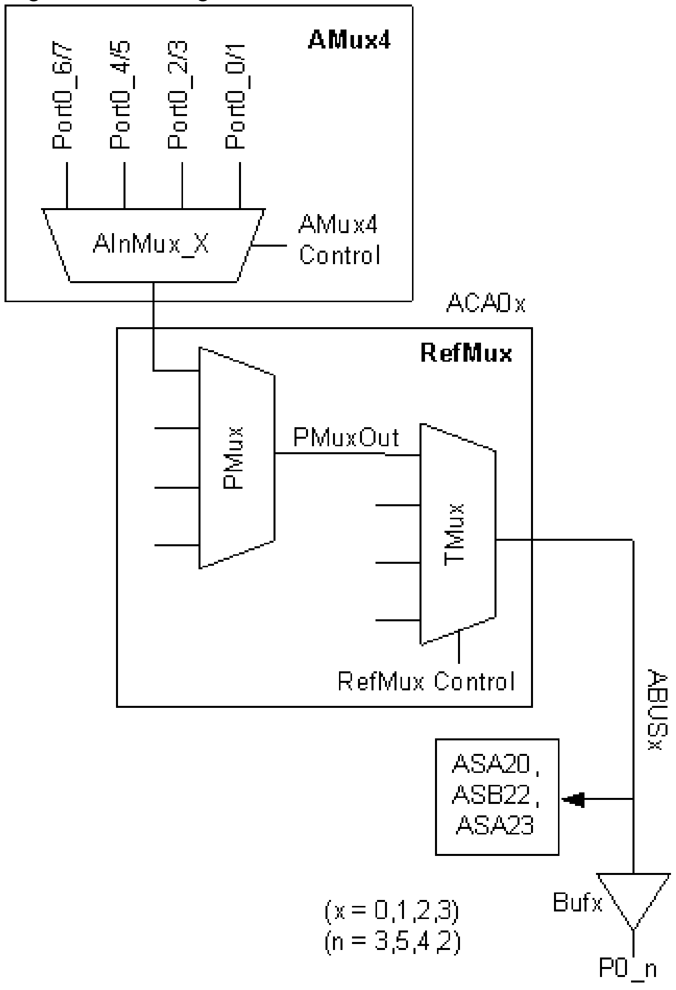
The AMux4 User Module maps into any of the AInMux blocks, as shown in Figure 3:

Figure 3. AMux4 Placement for the CY8C24/27xxx PSoC Devices



If used in conjunction with a RefMux User Module, the four input signals may be multiplexed onto the analog column bus. This allows routing rail-to-rail signals into some of the switched capacitor blocks (or user modules) such as filters and ADCs. The AMux4 User Module does not use a CT block, but connects to most user modules that do use CT blocks. Figure 4 shows a configuration with the AMux4 and a RefMux.

Figure 4. Usage of AMux and RefMux Combined



## Parameters and Resources

### Analog Column Mux

This parameter selects which multiplexer is selected. Valid inputs are between 0 and 3 for the four analog columns. The far left column is ‘0’ and the far right column is ‘3’. The multiplexer used for this user module is the AInMux_n mux, directly above the CT block in the selected analog column.

## Application Programming Interface

The Application Programming Interface (API) routines are provided as part of the user module to allow the designer to deal with the module at a higher level. This section specifies the interface to each function together with related constants provided by the “include" files.

### Note

In this, as in all user module APIs, the values of the A and X register may be altered by calling an API function. It is the responsibility of the calling function to preserve the values of A and X before the call if those values are required after the call. This “registers are volatile" policy was selected for efficiency reasons and has been in force since version 1.0 of PSoC Designer. The C compiler automatically takes care of this requirement. Assembly language programmers must ensure their code observes the policy, too. Though some user module API functions may leave A and X unchanged, there is no guarantee they will do so in the future.

For Large Memory Model devices, it is also the caller's responsibility to preserve any value in the CUR_PP, IDX_PP, MVR_PP, and MVW_PP registers. Even though some of these registers may not be modified now, there is no guarantee that will remain the case in future releases.

## AMUX4_InputSelect

**Description:**

Switches selected port to the CT block.

**C Prototype:**

```
void  AMUX4_InputSelect(BYTE bChannel);
```

**Assembly:**

```
mov   A, AMUX4_PORT0_3
lcall  AMUX4_InputSelect
```

**Parameters:**

bChannel: One byte that specifies which port pin will be connected to the CT Block. The usable ports are dependent on which column the AMUX4 User Module is placed. If the AMUX4 module is placed in column 0 or 2, Port0 pins 1, 3, 5 or 7 are selectable. Placing the AMUX4 module in column 1 or 3, allows selection of Port0 pins 0, 2, 4, or 6. Symbolic names provided in C and assembly, and their associated values, are listed in the following table:

| Symbolic Name | Value |
|---|---|
| AMUX4_PORT0_0 | 0x00 |
| AMUX4_PORT0_2 | 0x01 |
| AMUX4_PORT0_4 | 0x02 |
| AMUX4_PORT0_6 | 0x03 |
| AMUX4_PORT0_1 | 0x00 |
| AMUX4_PORT0_3 | 0x01 |
| AMUX4_PORT0_5 | 0x02 |
| AMUX4_PORT0_7 | 0x03 |

**Note** The symbolic names are generated regardless of whether the device you are using has these ports available or not. Do not use symbolic names associated with pins that do not exist on your device.

**Return Value:**

None

**Side Effects:**

The A and X registers may be altered by this function.

### AMUX4_Start

**Description:**

Currently this function performs no action and is not required for operation of the AMux4. It is provided for compatibility only.

**C Prototype:**

```
void  AMUX4_Start(void);
```

**Assembly:**

```
lcall  AMUX4_Start
```

**Parameters:**

None

**Return Value:**

None

**Side Effects:**

The A and X registers may be altered by this function.

### AMUX4_Stop

**Description:**

Currently this function performs no action and is not required for operation of the AMux4. It is provided for compatibility only.

**C Prototype:**

```
void  AMUX4_Stop(void);
```

**Assembly:**

```
lcall  AMUX4_Stop
```

**Parameters:**

None

**Return Value:**

None

**Side Effects:**

The A and X registers may be altered by this function.

## Sample Firmware Source Code

Here is a simple assembly and C example for selecting an analog signal using the AMux4:

```
;;-----------------------------------------------------------------
;; Sample Code for the AMUX4 user module
;; In this example, the AMUX4 user module is placed in column 2.
;; This allows Port0 pins 1,3,5,7 to be connected to the
;; ACB02 CT block.
;;-----------------------------------------------------------------

export _main
```

```
include "m8c.inc"
include "AMUX4.inc"

_main:

    mov  A,AMUX4_PORT0_3       ; specify port pin Port0_3
    call AMUX4_InputSelect     ; to connect to the CT block

    ; …Other code
    ret
```

A sample project written in C is:

```
//-------------------------------------------------------------------
// Sample Code for the AMUX4 user module
// In this example, the AMUX4 user module is placed at location ACA02,
// column 2.  This allows Port0 pins 1,3,5,7 to be connected to the
// ACB02 CT block.
//-------------------------------------------------------------------


#include "m8c.h"
#include "PSoCAPI.h"

void main(void)
{
   BYTE bPortNumber;

bPortNumber = AMUX4_PORT0_3;         // Assign port number
   AMUX4_InputSelect(bPortNumber);      // Switch Port0, pin3 to the
                                  // CT block.


   // …Other code
}
```

## Configuration Registers

This register is configured by the initialization and API library. You do not have to change or read this register directly. This section is given as a reference.

Table 3. Register AMX_IN

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Value | ACI3[1] | ACI3[0] | ACI2[1] | ACI2[0] | ACI1[1] | ACI1[0] | ACI0[1] | ACI0[0] |

ACI3[1:0]: Mux control bits for AInMux_3. ACI2[1:0]: Mux control bits for AInMux_2. ACI1[1:0]: Mux control bits for AInMux_1. ACI0[1:0]: Mux control bits for AInMux_0.

## Version History

| Version | Originator | Description |
|---|---|---|
| 1.4 | DHA | Added Version History |
| 1.50 | DHA | Corrected the pin mask to fix the pin connection to AMUX for CY8C28x13 parts. |

**Note** PSoC Designer 5.1 introduces a Version History in all user module datasheets. This section documents high level descriptions of the differences between the current and previous user module versions.