PIO
Driver

Rafael Corsi Ferrão
corsiferrao@gmail.com

31 de março de 2016

1 Input

Podemos configurar os pinos comandados pelo PIO como sendo uma entrada digital, essas entradas podem representar diversos sinais de controle, tais como :

- Fim de curso de um trilho;
- Botão para interface com o usuário;
- Contagem de eventos;
- ...

1 - Entrada digital

De um exemplo de um sinal digital que pode ser utilizado em um projeto de eletrônica embarcada.

1.1 Configurando o SAM4S

Do manual, página 574.

31.5.6 Inputs

The level on each I/O line can be read through PIO_PDSR. This register indicates the level of the I/O lines regardless of their configuration, whether uniquely as an input, or driven by the PIO Controller, or driven by a peripheral.

Reading the I/O line levels requires the clock of the PIO Controller to be enabled, otherwise PIO_PDSR reads the levels present on the I/O line at the time the clock was disabled

Pelo texto verificamos que o registrador PIO_PDSR contém o valor lido no determinado pino, porém para seu correto funcionamento é necessário primeiramente ativar o PIO para controlar o pino em questão (PIO_PSR), devemos por último desativar o buffer de saída via registrador PIO_ODR, tornando o valor do pino controlado externamente ao uC.

Um resumo das configurações dos registradores pode ser visto na Fig. 1, onde em vermelho os registradores que devem ser colocados em 0, em verde os registradores que devem ser ativados em 1 e em marrom os registradores opcionais.

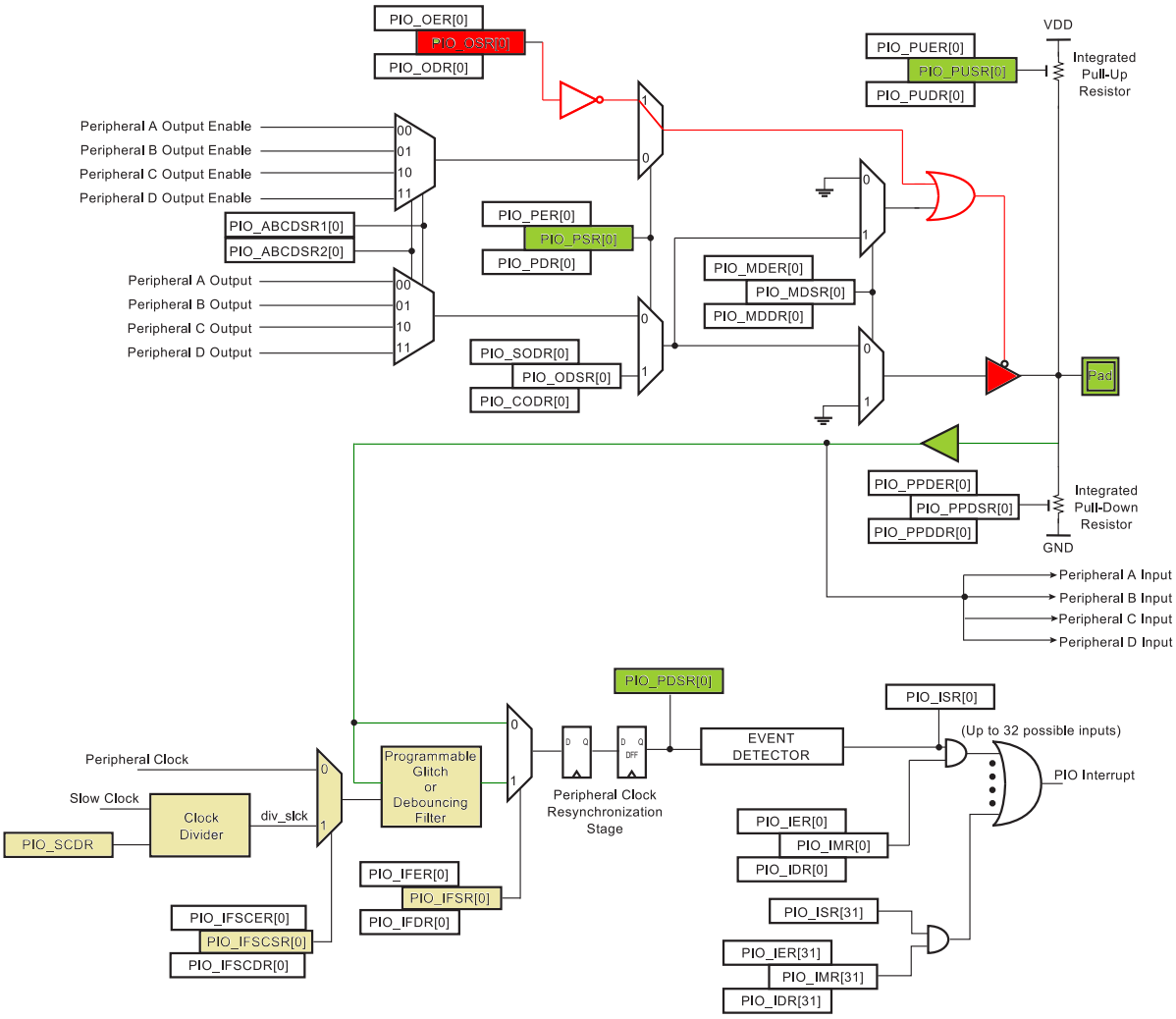


Figura 1: Registradores e fluxo de dado

1.2 Pull-UP/ Pull-Down

Os registradores PIO_PUSR e PIO_PPDSR controlam respectivamente a ativação do *pull-up* e *pull-down* do respectivo pino.

2 - Valores resistores

Qual o valor dos resistores de *pull-up* e *pull-down* ?

1.2.1 KIT

Isso é necessário pois o kit não possui um resistor conectado ao botão.

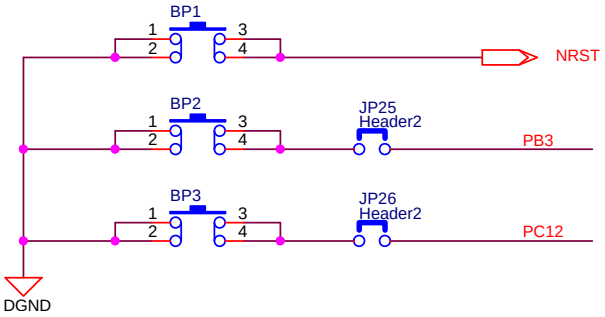


Figura 2: Esquemático SAM4S-EK2

2.1 - Valor Lido

Qual o valor lido pelo PIO quando o botão não estiver pressionado e qual o valor lido quando o botão estiver pressionado ?

1.3 Debouncing + Glitch

O registrador PIO_IFSR seleciona se utilizaremos o hardware dedicado para *deboucing* e filtro de *glitch*, se o valor selecionado for 0 (pelo PIO_IFDR), o sinal lido pelo I/O não passa por esse hardware específico e o valor lido deve ser tratado por software. Já se o valor for configurado como verdadeiro (pelo PIO_IFER), podemos selecionar o modo de tratamento desse sinal.

- se PIO_IFSCSR[i] = 0 : Filtro de glitch é ativado
- se PIO_IFSCSR[i] = 1 : Deboucing é ativado

Esse bloco de hardware opera com um clock menor que o do PIO, a redução de frequência é controlada por um registrador (PIO_SCDR) que divide a frequência de

entrada do periférico pelo valor posto nesse registrador de 32 bits. A fórmula utilizada para a obtenção do novo clock é :

$$t_{div_slck} = ((PIO\_SCDR + 1) * 2) * t_{slck}$$

Onde t_{div_slck} é o período do clock resultante e t_{slck} é o período do clock de entrada (original).

3 - Divisor de clock

Qual o valor máximo que PIO_SCDR pode assumir ?
Quando (PIO_SCDR for zero, por quanto o clock principal é dividido ?

4 - Interpretação carta de tempo

Interprete os diagramas de tempo a seguir (referentes ao filtro de *glitch* e *deboucing*).

Figure 31-4. Input Glitch Filter Timing

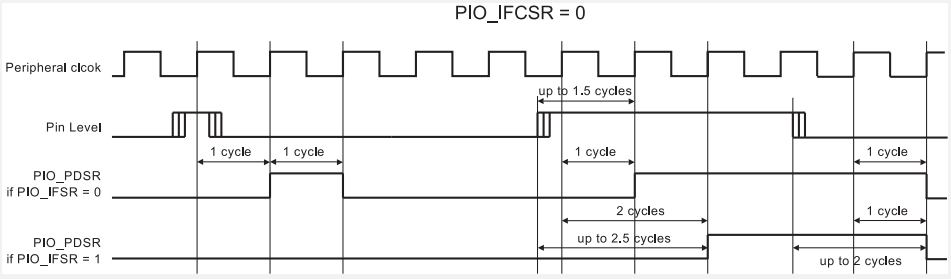
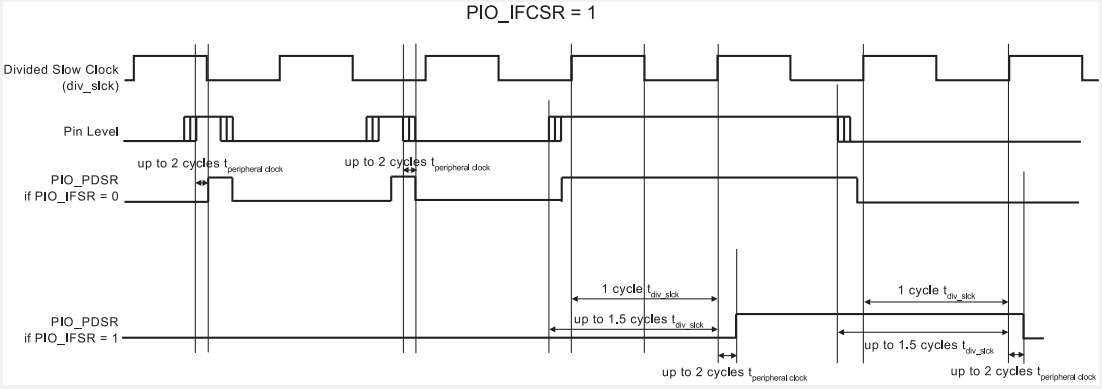


Figure 31-5. Input Debouncing Filter Timing



2 Programação

Desenvolva um programa que através de um botão controle o piscar dos LEDs (start/stop). O programa pode ser a evolução do desenvolvido na aula passada.

1. Identifique o PIO e o pino que pode ser acionado pelos botões
2. Ative o clock do PIO (se necessário)
3. Ative o PIO para controlar o pino do botão
4. **Desative o buffer de saída**

   Importante, caso contrário pode danificar o uC!

   5 - Entrada

   O que pode acontecer caso configuremos o pino do botão como saída ? Explique.

5. Ative o pull-up
6. Decida se irá ativar ou não o deboucing
   - não ativado : deve tratar o problema por sw
   - ativado : deve configurar os registradores (mais indicado)
7. faça a leitura periódica no while(1) para checar se o botão foi ou não pressionado

Não esqueça de dar commits no git para mantermos um histórico da evolução do código.

6 - While(1)

Qual a alternativa para evitar que o status do botão seja (ou precise ser) verificado continuamente?