Como mencionado na seção anterior, o *DriverLib* é um conjunto de APIs utilizado para controlar, configurar e manipular os periféricos do microcontrolador. Além de deixar o código mais intuitivo, esse pacote simplifica a criação de um código de fácil portabilidade para plataformas da família MSP432 e MSP430.

Utilizar o *DriverLib*  como camada de HAL é vantajoso em virtude de ser uma solução testada por profissionais e bem documentada. O Guia do Usuário do DriverLib apresenta um exemplo onde compara a configuração do *MasterClock* utilizando o nível de registradores, Figura X, e o API-*CS\_initClockSignal()*, Figura X+1. Com esse exemplo fica evidente o grau de abstração e facilidade que o *DriverLib* oferece.



Figura X A



Figura XB

A tabela abaixo mostra as vinte e cinco APIs do pacote e uma breve descrição sobre cada uma. Caso o leitor queira se aprofundar na leitura poderá consultar a referência asdas

|  |  |
| --- | --- |
| **API** | **Descrição** |
| ADC4 | Permite o controlar os conversores Analógico Digital. |
| AES256 | Permite a criptografia e descriptografia de dados de 128bits, de acordo com o padrão (AES256) |
| COMP\_E | Essa API fornece um conjunto de funções para inicializar os módulos COMP\_E, de comparação de dois sinais de entrada analógicos. |
| CRC32 | Permite fornecer um conjunto de funções para a verificação de dados. Essas funções são uteis quando há a necessidade de verificar a acurácia de um dado recebido em um canal de comunicação. |
| CS | Permite controlar o sistema de clock do microcontrolador. |
| DMA | Essa API permite controlar o *Direct Memory Access* (DMA) do microcontrolador, permitindo transferir blocos de dados sem a necessidade de utilizar o processamento do microcontrolador. |
| FlashCtl | Permite o controlar o processo de gravar, apagar e configurar a memória interna do processador. |
| FPU | Essa API fornece métodos para manipular o comportamento da Unidade de Ponto Flutuante (FPU do inglês *Floating-Point Unit*) do processador Cortex-M. |
| GPIO | Permite configurar e ativar os pinos de entrada/saída do microcontrolador e configurar as interrupções. |
| I2C | Essa API fornece o controle do barramento eUSCI\_B, no modo I2C, permitindo a configuração da frequência de transmissão, envio/recebimento de dados, status, etc. |
| NVIC | Permite controlar *o Nested Vectored Interrupt Controller* (NVIC). Esse modulo ativa, desativa, registra e configura as prioridades das interrupções do microcontrolador, |
| MPU | Essa API fornece funções para configurar o *Memory Protection Unit* (MPU). O MPU é acoplado ao núcleo do processador Cortex-M e fornece um meio de estabelecer permissões de acesso à regiões da memória. |
| PCM | Permite o gerenciamento dos estados de energia do microcontrolador. |
| PMAP | Essa API permite configurar o modulo *Port Mapping Controller.* Esse modulo é responsável por reconfigurar as funções digitais de cada porta. |
| PSS | Permite a configuração das várias entradas de alimentação do microcontrolador, de modo a otimizar a eficiência energética. |
| REF\_A | Permite configurar e ativar o uso da tensão de referência REF\_A |
| ResetCtl | Permite configurar e manipular as funções do reset do microcontrolador, tanto soft reset quanto hard reset. |
| RTC\_C | Essa API fornece um conjunto de funções para controlar o Real Time Clock (RTC\_C). |
| SPI | Essa API fornece o controle do barramento eUSCI\_A/eUSCI\_B, no modo SPI, permitindo a configuração da frequência de transmissão, envio/recebimento de dados, status, etc. |
| SysCtl | Essa API junta os módulos do sistema que não se encaixam em nenhum periférico especifico. |
| SysTick | O SysTick é um temporizador simples que fornece uma interrupção periódica para RTOS, mas ele pode ser usado para outros fins de temporização. |
| Timer32 | Permite a configuração do Timer32 (contador de 32 bits). |
| Timer\_A | Essa API permite configurar o modulo *TimerA*. Esse modulo é um temporizador/contador de 16 bits, suportando múltiplos modos captura/comparação, PWM e temporização de intervalos. |
| UART | Essa API fornece o controle do barramento serial USCI, permitindo a configuração da frequência de transmissão, envio/recebimento de dados, status, etc. |
| WDT\_A | Permite o controle do *Watchdog* padrão do sistema. |