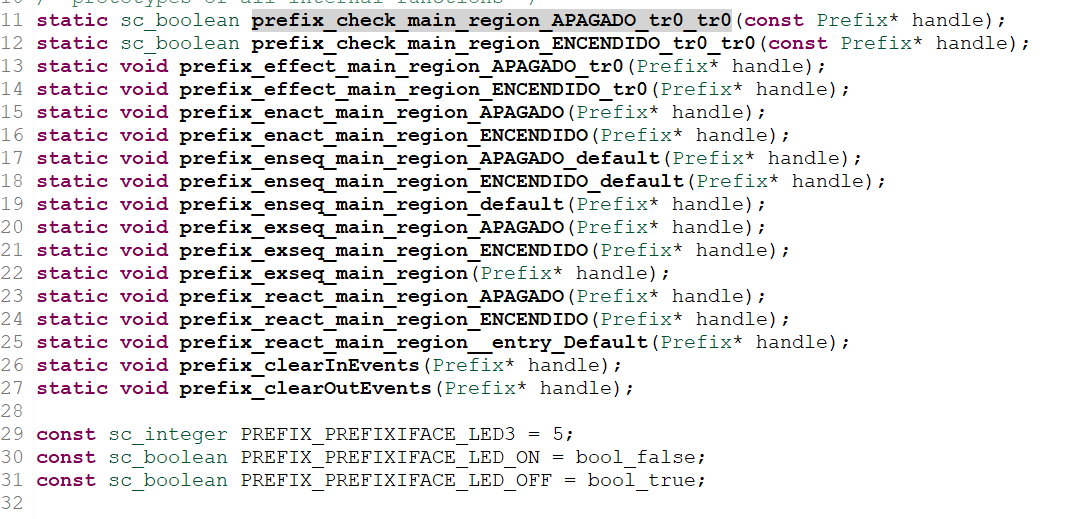
En primer lugar, se analizarán los tipos de datos de las funciones y constantes declaradas en el archivo

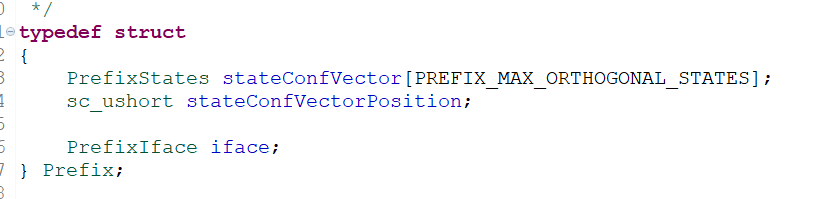
firmware\_v2\sapi\_examples\edu-ciaa-nxp\statecharts\statecharts\_bare\_metal\gen\Prefix.c



En el archivo

firmware\_v2\sapi\_examples\edu-ciaa-nxp\statecharts\statecharts\_bare\_metal\gen\Prefix.h

se encuentra definido “Prefix” como la siguiente estructura:



Donde “PrefixStates” es un enum con los distintos posibles estados de la región:

**typedef** **enum**

{

*Prefix\_main\_region\_APAGADO*,

*Prefix\_main\_region\_ENCENDIDO*,

*Prefix\_last\_state*

} PrefixStates;

Y “PrefixIface” es una estructura con un booleano:

**typedef** **struct**

{

sc\_boolean evTick\_raised;

} PrefixIface;

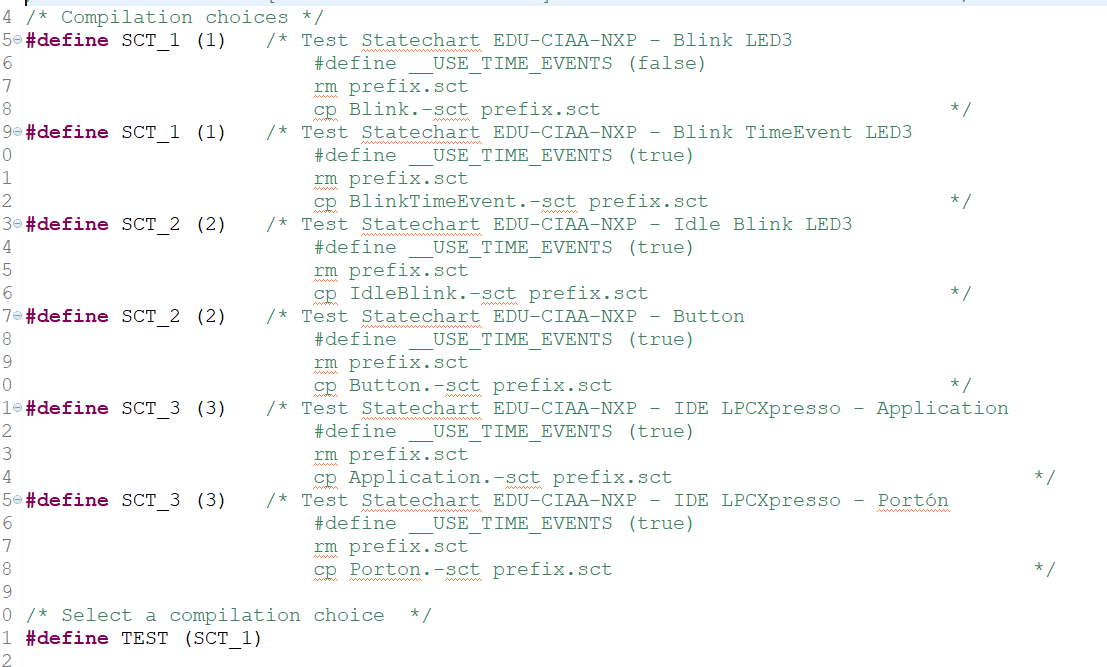
La variable stateConfVectorPosition es un int para guardar la posición del vector.

Estos tres datos son los que reciben las funciones declaradas y los que se manipulan dentro de las mismas, por ejemplo, para ir cambiando de estado.

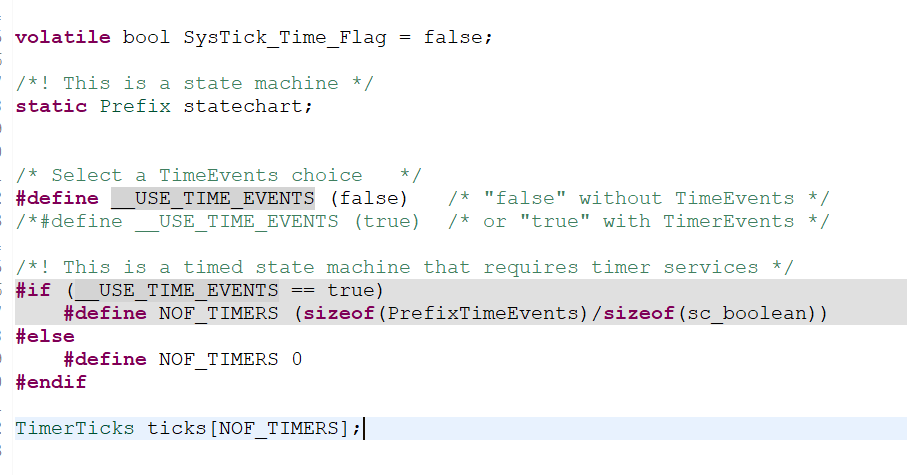
A continuación, se analizarán los tipos de datos de las funciones y constantes declaradas en el archivo

/firmware\_v2/sapi\_examples/edu-ciaa-nxp/statecharts/statecharts\_bare\_metal/src/main.c

Al principio del archivo se definen las opciones de compilación y se selecciona qué ejemplo se quiere ejecutar



Luego se define un flag de tipo volátil, una variable de tipo Prefix (ya explicada), el uso de los TIME\_EVENTS y una variable de tipo TimerTicks, que es una estructura que se encuentra declarada en /firmware\_v2/examples/statechart/inc/TimerTicks.h



**typedef** **struct**

{

sc\_eventid evid;

sc\_uinteger time\_ms;

sc\_uinteger count;

sc\_boolean periodic;

sc\_boolean active;

sc\_boolean evPending;

} TimerTicks;

El resto del programa (cuya secuencia de funciones está explicada en otro documento) usa principalmente los datos nombrados anteriormente.