1. What is unit testing, functional testing, module testing, integration testing?
2. We have CRC, why we go for CMAC?
3. What is checksum?

A checksum is a count of the number of bits in a transmission unit that is included with the unit so that the receiver can check to see whether the same number of bits arrived.

1. What is autosar, why is it useful?
2. Difference between checksum and CRC?
3. What is pre compile, post build, link time ?

* **Pre-compile configuration** − Configuration parameters can not be changed after compilation − Example: Mapping of microcontroller pins to signals
* **Link-time configuration** − Configuration is determined by linker scripts − Configuration parameters can not be changed after link process − Purpose: provides capability to deliver object code to the integrator
* **Post-build configuration** –

1. **Post-build time loadable**

--Configuration parameters can be changed after build process without complete re-flash of ECU

1. **Post-build time selectable**

--Configuration parameter set is selected from multiple configuration sets during boot time

-- All possible configuration sets need to be included at compile time

-- Configuration parameters are stored at a known memory location

--Post-build configuration class BSW modules might also contain pre-compile or link-time parameters (not all parameters have to be post-build)

--Purpose: use one software package in different vehicle

1. What is the difference between compiling and linking?

**Compilation** refers to the processing of source code files (.c, .cc, or .cpp) and the creation of an 'object' file.  if you compile (but don't link) three separate files, you will have three object files created as output, each with the name <filename>.o or <filename>.obj (the extension will depend on your compiler). Each of these files contains a translation of your source code file into a machine language file -- but you can't run them yet! You need to turn them into executables your operating system can use. That's where the linker comes in.

**Linking** refers to the creation of a single executable file from multiple object files. In this step, it is common that the linker will complain about undefined functions (commonly, main itself). During compilation, if the compiler could not find the definition for a particular function, it would just assume that the function was defined in another file. If this isn't the case, there's no way the compiler would know -- it doesn't look at the contents of more than one file at a time. The linker, on the other hand, may look at multiple files and try to find references for the functions that weren't mentioned.

Gopi