Attaining functional safety - Hazard analysis and risk assessment

Hoiman Low, Microcontroller Business Development Manager, Texas Instruments (TI) - March 07, 2016

Products being used within their end-product application can suffer systematic or random failures. Functional safety standards are designed to help influence the reduction of potential risks of physical injury to people and property damage due to such failures. But first these risks need to be identified for the end product application, and the effects of such risks analyzed so that appropriate measures can be implemented to reduce both the likelihood of their occurrence and – if and when they do occur – their impact from an overall safety perspective.

Product failure modes and failure rates need to be understood and estimated. In my last article, we examined how to manage random failures. This blog will examine hazard analysis and risk assessment at the product level to help determine the product risk reduction requirement -- ASIL/SIL -- as well as the required hardware metrics need to measure risk reduction. Once a requirement is known, developers will need to implement enough safety mechanisms (diagnostics) to detect random failures in order to achieve sufficient risk reduction, as measured by the required metrics indicated in the applicable standard, for each safety function.

Let's use an Electric Vehicle (EV) traction motor control Electronic Control Unit (ECU) as an example to illustrate the flow and activities involved, from the hazard analysis and risk assessment to the failure rate metrics estimation (See Figure 1). For this example, the appropriate standard is ISO 26262. Similar methodologies will apply with other end product functional safety standards as well.
The EV traction motor control ECU’s function is to provide appropriate torque to the motor, based on the driver input. One example of a failure mode is the application of too high a positive torque to the motor, which can occur on a city road or during highway driving conditions. This hazard is an important consideration because it raises the possibility of frontal collision with life-threatening injuries, a critical application care-about for the end product developer.

According to ISO 26262-3:2011, hazard analysis and risk assessment is to be performed based on malfunctioning behavior of a system and by evaluating for each behavior the following: severity (S1-S3), probability of exposure (E1-E4), and controllability (C1-C3).

In this example the evaluation is:

- Severity (S) = S3 because of life-threatening injuries;
- Probability of Exposure (E) = E4 because of the high probability of driving on a city road or highway conditions; and
- Controllability (C) = C2 because of controllability by braking.

Based on the ISO 26262 ASIL determination table (Figure 2), this hazardous event would most likely be classified as ASIL-C, because the functional safety goal is to avoid too high positive torque applied to the motor.
With ASIL-C, the risk reduction requirements at the item (product) level, as measured by the required hardware metrics, are listed in Table 1 below.

### Table 1

<table>
<thead>
<tr>
<th>ASIL</th>
<th>SPFM</th>
<th>PMHF (FIT)</th>
<th>LFM</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASIL B</td>
<td>&gt;90%</td>
<td>&lt;100</td>
<td>&gt;60%</td>
</tr>
<tr>
<td>ASIL C</td>
<td>&gt;97%</td>
<td>&lt;100</td>
<td>&gt;80%</td>
</tr>
<tr>
<td>ASIL D</td>
<td>&gt;99%</td>
<td>&lt;10</td>
<td>&gt;90%</td>
</tr>
</tbody>
</table>

Table 1. SPFM is a single point fault metric, PMHF is a probabilistic metric for random hardware failures and LFM is a latent fault metric.

In this example, let’s assume the motor torque is commanded by another ECU in the vehicle via CAN. The microcontroller (MCU) will control the motor torque by appropriately actuating PWMs.
We will need, then, to understand how the PWM signals are generated and the possible MCU failure modes that could impact them, and then calculate the failure rate. If the failure rate is too high versus the ASIL-C requirement, then diagnostics will need to be applied to reduce the failure rate. After the implementation of such a diagnostic, the diagnostic coverage and the residual failure rate should be re-evaluated to see if sufficient risk reduction is achieved per applicable functional safety standard.

The simplified block diagram (Figure 3) shows the following:

- An external voltage regulator powers the MCU
- The MCU’s clock source is an external crystal
- The MCU receives motor torque command from a remote host via CAN
- Motor position is determined by a quadrature encoder and phase current from a bridge driver
- The CPU supported by the flash, and SRAM memory executes a Field Oriented Control (FOC) algorithm to determine the appropriate PWM signals to control the motor with desired torque
- The on-chip PWM timer then generates the PWM signals driving the bridge driver

Here is a sample list of MCU failures that could cause an incorrect PWM signal to be generated and sent to the bridge driver. Please note that this is by no means exhaustive and is only intended as an example for illustration purposes.

- CPU logic failure
- Flash and/or SRAM memory bit flip
- Power failure
- Clock failure
- CAN communication error
In our torque control example, these modules could each play an important role in implementing the motor torque control function, in which case malfunction of any of these modules could potentially cause a hazardous event to occur. To implement the safety function of “avoiding too high positive torque applied to the motor,” then, the designer would need to apply diagnostics to the safety-critical elements such that if any failure is detected, the system can take action to mitigate the failure or to transition to a well-defined safe state.

How would a system developer know what diagnostics are available and should be used for those elements within an MCU being used for safety-critical functions? I have written in previous blogs that one tool available to system developers is a safety manual that should be vendor-provided for any MCU intended for use in safety applications. An example is TI’s Hercules TMS570 MCU safety manual, which describes available diagnostic features for each module within the MCU, such as power, clock, CPU, flash, SRAM, PWM, CAN, etc.

Once the system developer can confidently confirm the safety elements used to implement the safety function and the applicable diagnostics for those safety elements, the next step for a system developer is to estimate the failure rate and to evaluate whether the required hardware metrics have been effectively achieved. As I mentioned in my previous blogs the Failure Modes Effect and Diagnostics Analysis (FMEDA) tool can help with this step. The FMEDA can be used to estimate the failure rate by safety function and the effect of risk reduction by applying diagnostics at the MCU level. Both permanent and transient failures are covered. This tool is highly customizable, allowing the user to tailor the analysis to its system operational profile, MCU resource and diagnostics usage, and the like. (The tool is offered, under NDA, with TI’s Hercules MCUs’ Detailed Safety Analysis Report and was jointly developed with our partner Yogitech.)

The FMEDA tool outputs the MCU ISO 26262 FIT rate estimates and safety metrics (SPFM, LFM, PMHF) for each safety function. Similar safety metrics are also available for IEC 61508. The MCU FIT rates and safety metrics can then be used in the product-level FMEDA to compute the product FIT rate and safety metrics.

Here, then, is the high-level overview of how to manage random hardware failures by a system developer:

- Define the product function and boundary
- Perform hazard analysis and risk assessment
- Determine safety integrity level and safety function(s)
- Identify failure modes and analyze failure rate by safety function(s)
- Apply diagnostics to detect failures until the fit rate and safety metrics requirements are met

The example used in this blog is based on an MCI-based motor control function, simply to illustrate hazard analysis and risk assessment flow. It is the system developer’s responsibility to perform their own hazard analysis and risk assessment at the system level as prescribed by the applicable functional safety standard for their own designs.

Safety documentation and tools offered by semiconductor vendors can help developers to perform random failure analysis at the product level, and can ease the developer’s efforts to achieve product verification.
Hoiman Low is TI’s functional safety MCU business development manager who has more than 30 years with TI and 20 years of experience in MCU design for functional safety applications.

Other blogs in this series:

Part 1: Attaining functional safety: Standards, certification, and the development process

Part 2: Attaining functional safety: Managing random failures