面向汽车安全的一键式 FMEDA 实现繁琐任务自动化

CHUCK BATTIKHA 和 DOUG SMITH，MENTOR, A SIEMENS BUSINESS

WHITEPAPER

Mentor®
A Siemens Business

MENTOR CONSULTING

www.mentor.com

摘要

汽车设计需要功能安全分析，此任务通常使用失效模式、影响与诊断分析 (FMEDA) 来完成。

FMEDA 用于确定每个安全目标的诊断覆盖率，进而确定设计是否满足目标 ASIL 要求。但是，如果您曾编写过 FMEDA，您便知道这是一项多么繁琐的任务。本文分享一种用于创建和自动执行 FMEDA 流程的一键式解决方案。

分析失效模式

ISO 26262 标准 [1] 要求对与安全相关的汽车 IC 设计进行定量分析。这种分析用于生成关键功能安全指标：PMHF[1]、SPFM[2] 和 LFM[3]。该标准根据汽车制造商期望 IC 设计满足的顶层系统 ASIL[4] 要求，为每个指标提供了目标。对于 PMHF，汽车制造商会将其系统 PMHF 的一部分分配给 IC 设计。为了确定这些指标，需要了解设计可能失效并导致危险（这是潜在危害或故障行为的来源[5]）的方式。标准和行业惯例是使用表格分析失效模式，在表格中列出每个设计组件的明细，并根据失效率或 FIT[6] 量化失效的影响。此表称为*失效模式、影响和诊断分析*，简称 *FMEDA*。

FMEDA 列出了每个组件以及每种失效模式对总体 FIT 率的贡献百分比。通过将安全机制纳入设计中，便可减轻失效模式的影响并降低其对 FIT 率的贡献。每种失效模式都要根据以下标准进行评估：是否与安全性相关，设计受失效模式影响的百分比，是否受到安全机制的保护，设计受该安全机制保护的百分比。表 1 定义了创建 FMEDA 涉及的常用术语。

有了这些信息，便可计算出失效模式的 FIT 率、残余／单点 FIT 率、安全 FIT 率、多点 FIT 率和潜在 FIT 率。例如，FMEDA 电子表格可以采取图 1 所示的形式。

[1] 随机硬件失效率。

[2] 单点故障度量参见 [1] 的第 5 部分 C.2。

[3] 潜在故障度量。参见 [1] 的第 5 部分 C.3。

[4] 汽车安全完整性级别。

[5] [1] 的第 1 部分第 3.88 条。

[6] 失效率：每十亿工作小时发生的元器件失效数量。

项目	描述
基本失效率 (FIT)	基本失效率（BFR，用 FIT 衡量）取决于工艺，是无任何保护情况下设计的固有 FIT。PMHF 要求可被视为目标 FIT。BFR 通常比设计的目标 FIT 大得多，因此要使用安全机制来保护设计并有效降低 BFR 以满足目标 FIT。在集成电路内，逻辑、模拟/混合信号电路和存储器具有不同的 BFR。假设寄存器和组合逻辑的 BFR 相同。永久失效和瞬态失效的 BFR 也会不同。在集成电路内，BFR 通常基于面积均匀分布在整个设计中。
安全目标	设计针对功能安全的顶层要求。集成电路中可能有多个安全目标。大多数集成电路不是以汽车为背景设计的，因此通常认为安全目标是为了确定与安全相关的内容以及失效是否会影响安全。
安全相关	设计组件是否与安全相关？它们是否会影响安全目标？它们是否会保护安全相关功能？通常，集成电路兼有安全相关逻辑和非安全相关逻辑。
失效模式	组件可能失效的所有方式。失效模式包括：生成的数据错误、提供的地址错误、在需要的时候未提供数据，等等。
失效模式分布	对于一个组件所识别的每种失效模式，发生该特定失效模式的百分比概率。
可能违反安全目标	失效是否可能违反安全目标？
安全机制	防止违反安全目标的所有已就位安全机制的清单。
诊断覆盖率	所确定的安全机制保护的失效模式百分比。使用所有安全机制的总和。

表 1：每种失效模式所需的 FMEDA 信息。

图 1：FMEDA 电子表格字段。

计算基本失效率

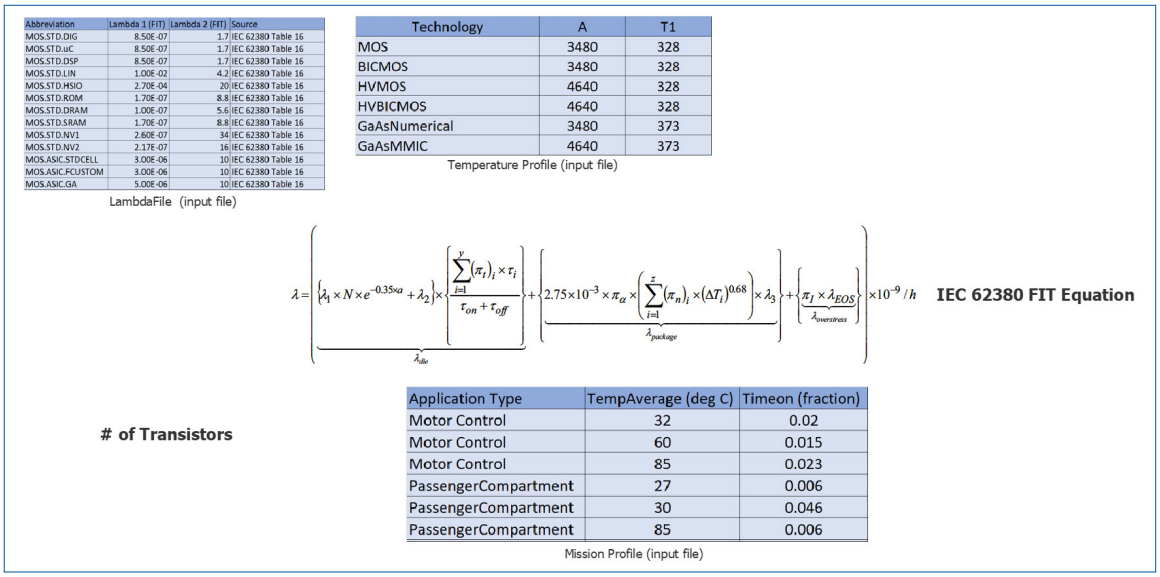
使用 FMEDA 可以确定设计的关键安全指标，但 FMEDA 也带来了相关挑战。第一个挑战是确定设计中使用的所有不同逻辑（标准单元、模拟、存储器、ROM、OTP 等）的基本失效率。ISO26262:2018-11 第 4.6 条详细介绍了基本失效率的估计。计算涉及许多希腊字母，其代表与不同技术相关的值。图 2 显示了用于计算基本失效率的示例工作表。

	A	B	C	D	E	F	G	H	I	J
1	**Die Base Failure Rate Calculation**									
2	**(Permenant Faults)**	Data source	Overall	Standard Cell	Analog	HSIO	SRAM	ROM	OTP	Notes
3	**Compute Raw Transistor FIT Rate**									
4	Lambda-1	26262:2018, pt 11, fig 10		1.20E-05	1.00E-02	2.70E-04	1.70E-07	1.70E-07	2.60E-07	
5	N (# of Transistors)	See "Physical Summary" sheet		4,118,800,193	7,079,288	423,030	1,485,223,296	2,162,688	294,912	SRAM transistors includes both High S Memory
6	% of Transistors	*Calculated*		73.37%	0.13%	0.01%	26.46%	0.04%	0.01%	
7	Mfg Year	Customer provided		2019	2019	2019	2019	2019	2019	
8	Year value	*Calculated*		21	21	21	21	21	21	
9	Exp	*Calculated*		0.000642592	0.000642592	0.000642592	0.000642592	0.000642592	0.000642592	
10	Raw Transistor FIT Rate	*Calculated*		31.76	45.49	0.07	0.16	0.00	0.00	
11	**Chip Mission Profile (time * temp)**									
12	**Compute Ton/Toff Mission Profile**									
13	pi1	*Calculated*	4.069039153							Based on T1 from Mission Table below
14	pi2	*Calculated*	3.168424174							T2
15	pi3	*Calculated*	1.839982568							T3
16	pi4	*Calculated*	0.848540291							T4
17	pi5	*Calculated*	0.117949743							T5
18	pi6	*Calculated*	0.117949743							T6
19	pi7	*Calculated*	0.117949743							T7
20	pi8	*Calculated*	0.117949743							T8
21	pi9	*Calculated*	0.117949743							T9
22	pi10	*Calculated*	0.117949743							T10
24	Pi1*Tao1	*Calculated*	0.03713474							Based on Tao1 and time % from Missi
25	Pi2*Tao2	*Calculated*	0.173493452							Tao2
26	Pi3*Tao3	*Calculated*	0.033583985							Tao3
27	Pi4*Tao4	*Calculated*	0.007743922							Tao4
28	Pi5*Tao5	*Calculated*	0							Tao5
34	Total Derating Factor	*Calculated*	0.2519561							
35	**Calulate FIT**									
36	ON/OFF Compensated Transistor FIT Rate (without Lambda2)	*Calculated*		8.002255	11.461726	0.018492	0.040879	0.000060	0.000012	Digital Logic has a defined lambda1 th SRAM. Digital Logic has 2.5x the num SRAM. Therefor Logic FIT is roughly 2 FIT.
37	Total FIT Rate without lambda2	*Calculated*	19.523425							
38	Lambda2	26262:2018, pt 11, fig 10		10	4.2	20	8.8	8.8	8.8	
39	Maximum Lambda2		20							The Maximum Lambda2 is used as su
40	Derating		0.2519561							
41	Lambda2 Adder		5.039121994							
42	FIT (lambda1 and lambda2)		24.5625							
43	Distributed lambda2			3.697042748	0.006354382	0.000379713	1.333139205	0.001941233	0.000264714	Lambdha2 distributed by Transistor Co
44	FIT by Logic & SRAM			**23.186251**			**1.376296**			
45										

图 2：用于计算基本失效率的工作表。

作为汽车功能安全顾问，我们手动创建了图 2 中的基本失效率表格，但我们一直在寻找一种更好的、更高效的方法来简化 FMEDA 流程，在这个方面 EDA 工具可以提供实际帮助。这种工具必须能够读入设计，确定其晶体管数量，测量每种安全机制的诊断覆盖率，并使用行业标准或自定义任务数据图表及技术值执行基本失效率计算，以自动计算每个设计组件的 FIT 率。

这些工具应允许我们使用从 IEC 62380 标准（ISO 26262 源自该标准）获得的默认值或由客户提供的输入值。工具确定逻辑和存储器的晶体管数量之后，将该值代入 IEC 62380 公式中，如图 3 所示。



Abbreviation	Lambda 1 (FIT)	Lambda 2 (FIT)	Source
MOS.STD.DIG	8.50E-07	1.7	IEC 62380 Table 16
MOS.STD.uC	8.50E-07	1.7	IEC 62380 Table 16
MOS.STD.DSP	8.50E-07	1.7	IEC 62380 Table 16
MOS.STD.LIN	1.00E-02	4.2	IEC 62380 Table 16
MOS.STD.HSIO	2.70E-04	20	IEC 62380 Table 16
MOS.STD.ROM	1.70E-07	8.8	IEC 62380 Table 16
MOS.STD.DRAM	1.00E-07	5.6	IEC 62380 Table 16
MOS.STD.SRAM	1.70E-07	8.8	IEC 62380 Table 16
MOS.STD.NV1	2.60E-07	34	IEC 62380 Table 16
MOS.STD.NV2	2.17E-07	16	IEC 62380 Table 16
MOS.ASIC.STDCELL	3.00E-06	10	IEC 62380 Table 16
MOS.ASIC.FCUSTOM	3.00E-06	10	IEC 62380 Table 16
MOS.ASIC.GA	5.00E-06	10	IEC 62380 Table 16

LambdaFile (input file)

Technology	A	T1
MOS	3480	328
BICMOS	3480	328
HVMOS	4640	328
HVBICMOS	4640	328
GaAsNumerical	3480	373
GaAsMMIC	4640	373

Temperature Profile (input file)

$$\lambda = \left\{ \underbrace{\left[\lambda_1 \times N \times e^{-0.35\times\alpha} + \lambda_2\right] \times \left\{ \frac{\sum_{i=1}^{y} (\pi_t)_i \times \tau_i}{\tau_{on} + \tau_{off}} \right\}}_{\lambda_{die}} + \underbrace{\left\{ 2.75\times10^{-3} \times \pi_\alpha \times \left(\sum_{i=1}^{z} (\pi_n)_i \times (\Delta T_j)^{0.68} \right) \times \lambda_3 \right\}}_{\lambda_{package}} + \underbrace{\left\{ \pi_I \times \lambda_{EOS} \right\}}_{\lambda_{overstress}} \right\} \times 10^{-9}/h$$

IEC 62380 FIT Equation

# of Transistors

Application Type	TempAverage (deg C)	Timeon (fraction)
Motor Control	32	0.02
Motor Control	60	0.015
Motor Control	85	0.023
PassengerCompartment	27	0.006
PassengerCompartment	30	0.046
PassengerCompartment	85	0.006

Mission Profile (input file)

图 3：IEC 62380 FIT 计算。

FIT 计算报告简化了每个设计组件的 FMEDA 计算（图 4）。

	A	B	C	D	E	F	G	H	I	J
1	*ModuleName	LibType	LambdaPermanent	LambdaTransient	DiscountedLambda Permanent	DiscountedLambda Transient	DiagCoveragePerm	DiagCoveragTran	LambdaTech	Nval
2	or1200_dmmu_top	CUST.STD.SRAM	2.35758	0.002432	2.35758	0.002432	0.6	0.6	21.4077	2432
3	or1200_dmmu_top	CUST.ASIC.STDCELL	1.40999	0.02	1.09687	0.001	0.363654	0.730769	10.7038	2909

图 4：FIT 计算报告。

固定和瞬态失效模式的 λ 值[7]及诊断覆盖率[8]直接代入 ISO26262:2018-5 第 C.3 条所述的 SPF/RF FIT 率公式中：

理想情况下，工具应自动确定适用于每个设计组件的晶体管的百分比；因此，图 4 所示的 λ 值已经考虑了组件对基本失效率 (BFR) 的贡献或面积百分比，如图 5 所示。

$$\lambda_{RF} = 1 - \frac{DC}{100} * \lambda_{\%\ of\ BFR}$$

图 5：用于计算 SPF 和 RF 失效的 λ 的公式。

[7] LambdaPermanent 和 LambdaTransient。

[8] DiagCoveragePerm 和 DiagCoverageTran。

加快速度！

现在，我们已经消除了确定基本失效率的第一个挑战，接下来必须应对执行所有 FIT 计算以创建 SPFM、LFM 和 PMHF 的艰巨任务。更具挑战性的是，集成电路中通常有数百个模块，导致 FMEDA 电子表格有成百上千行，以显示所有需要标注的模块或组件及其各种失效模式（参见图 6）。

	A	B	D	E	F	G	H	I	J	K	L
1						1,006,048.80	11,717,316	100.0%	100.0%	0.284	0.693
2	Sub-Part (Reference)	Sub-Part (Instance Name)	Effect	Safety Related Block (Y/N)?	Is Sub-Part a Safey Mechanism (Y/N)?	Area [um2]	Area [G/C]	Area % [um2]	Area % [G/C]	Sub-Part λ (FIT) PERM	Sub-Part λ (FIT) TRANS
3	[redacted]	[redacted]	Not Safety Related	N	N	46.95	547	0.0%	0.0%	0.00001	0.00003
4	[redacted]	[redacted]	1. Corruption of data in a transfer (wrong data in write	Y	N	1,365.85	15,908	0.1%	0.1%	0.00039	0.00094
5	[redacted]	[redacted]	1. Corruption of data in a transfer (wrong data in write	Y	N	324.04	3,774	0.0%	0.0%	0.00009	0.00022
6	[redacted]	[redacted]	1. Corruption of data in a transfer (wrong data in write	Y	N	200.20	2,332	0.0%	0.0%	0.00006	0.00014
7	[redacted]	[redacted]	1. Corruption of data sent to SYSREG (leads to incorr	Y	N	4,605.32	53,638	0.5%	0.5%	0.00130	0.00317
8	[redacted]	[redacted]	1. Corruption of data in a transfer (wrong data in write	Y	N	537.89	6,265	0.1%	0.1%	0.00015	0.00037
9	[redacted]	[redacted]	1. Corruption of data in a transfer (wrong data in write	Y	N	12,688.67	147,783	1.3%	1.3%	0.00358	0.00874
10	[redacted]	[redacted]	1. No clock being generated 2. Wrong clock frequenc	Y	N	7,618.03	88,726	0.8%	0.8%	0.00215	0.00525
11	[redacted]	[redacted]	1. Compare (alarm) always firing 2. Compare (alarm)	Y	Y	37.77	440	0.0%	0.0%	0.00001	0.00003
12	[redacted]	[redacted]	Not Safety Related	N	Y	37.77	440	0.0%	0.0%	0.00001	0.00003
13	[redacted]	[redacted]	1. Compare (alarm) always firing 2. Compare (alarm)	Y	Y	38.65	450	0.0%	0.0%	0.00001	0.00003
14	[redacted]	[redacted]	1. Compare (alarm) always firing 2. Compare (alarm)	Y	Y	43.00	501	0.0%	0.0%	0.00001	0.00003
15	[redacted]	[redacted]	Not Safety Related	N	Y	38.65	450	0.0%	0.0%	0.00001	0.00003
16	[redacted]	[redacted]	1. CRC generated incorrectly	Y	Y	1,467.55	17,092	0.1%	0.1%	0.00041	0.00101
17	[redacted]	[redacted]	1. CRC generated incorrectly	Y	Y	1,467.55	17,092	0.1%	0.1%	0.00041	0.00101
18	[redacted]	[redacted]	1. Secure Traffic is blocked when it should be allowed	Y	N	197.12	2,296	0.0%	0.0%	0.00006	0.00014
19	[redacted]	[redacted]	Not Safety Related	N	N	163.84	1,908	0.0%	0.0%	0.00005	0.00011
20	[redacted]	[redacted]	Not Safety Related	N	N	153.51	1,788	0.0%	0.0%	0.00004	0.00011
21	[redacted]	[redacted]	1. Corruption of data transfer	Y	N	336.83	3,923	0.0%	0.0%	0.00009	0.00023
22	[redacted]	[redacted]	1. Corruption of data in a transfer (wrong data in write	Y	N	2,108.10	24,553	0.2%	0.2%	0.00059	0.00145
23	[redacted]	[redacted]	1. No clock being generated 2. Glitches on clocks (do	Y	N	94.64	1,102	0.0%	0.0%	0.00003	0.00007
24	[redacted]	[redacted]	1. Spurious Interrupts 2. Continuous Interrupts 3. No I	Y	Y	14,478.09	168,624	1.4%	1.4%	0.00408	0.00997
25	[redacted]	[redacted]	Not Safety Related	N	N	14,478.09	168,624	1.4%	1.4%	0.00408	0.00997
26	[redacted]	[redacted]	Not Safety Related	N	N	1,583.77	18,446	0.2%	0.2%	0.00045	0.00109
27	[redacted]	[redacted]	1. Corruption of data transfer	Y	N	633.51	7,378	0.1%	0.1%	0.00018	0.00044
28	[redacted]	[redacted]	Not Safety Related	N	N	127,436.38	1,484,234	12.7%	12.7%	0.03593	0.08775
29	[redacted]	[redacted]	Not Safety Related	N	N	127,436.38	1,484,234	12.7%	12.7%	0.03593	0.08775
30	[redacted]	[redacted]	Not Safety Related	N	N	127,436.38	1,484,234	12.7%	12.7%	0.03593	0.08775

图 6：有数百个组件要分析的 FMEDA。注：对 A 列和 B 列中的数据进行了模糊处理，以保护专有信息。

因此，第二个挑战便是找到一种更简单且自动化的方法来直观地创建 FMEDA 电子表格，使用户可以遍历设计，选择与安全相关且需要分析的组件，并使用 FIT 率执行所有必要的计算。应该注意的是，FMEDA 是 ISO 26262 安全分析要求的产物。因此，汽车制造商及其一级供应商会在 FMEDA 中寻找细节来展示和证明其所取得的成果。仅仅标注一个最终数字是不够的。这类似于高中时候的考试，您需要展示您的全部实力。

作为顾问，我们要承担繁琐的工作，在 FMEDA 中创建成百上千行数据。FMEDA 倾向于以电子表格展示，因此我们决定使用 Microsoft ®Excel 内置的 VisualBasic 自动执行该过程。FMEDA 创建部分是在 Windows 平台上的 Excel 中执行的，但设计提取和基本失效率／诊断覆盖率计算是在 Linux 中使用行业标准 RTL 仿真器（例如 Mentor, A Siemens Business 的 Questa®）和基本失效率计算工具来执行的。许多公司将项目目录安装在 Windows 台式机上，使该过程无缝进行，但如果项目目录不可远程安装，文件也可以轻松地来回复制 (scp)。（我们还使用 Cygwin 修复了关于已安装网络驱动器的权限问题。）

虽然可以直接调用宏，但我们在电子表格中创建了一个交互式界面，支持轻松输入 FMEDA 流程的所有设置信息，如图 7 所示。其中提供了网络映射以在 Windows 和 Linux 之间转换路径。

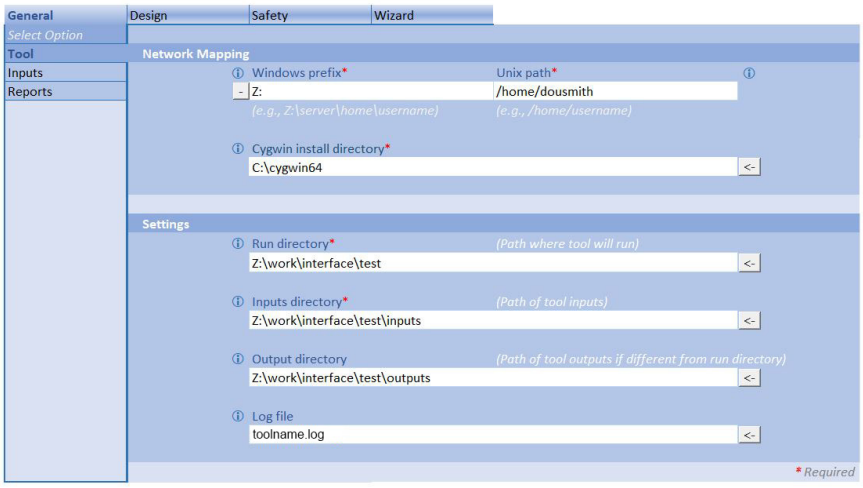


图 7：基于 Excel 的界面，用于输入 FMEDA 设置信息。

第一步是指定有关设计的所有安全信息，例如：任务数据图表、技术λ值、不同设计元件基本单元的晶体管数量，以及特定安全机制的诊断覆盖率信息（参见图 8 和图 9）。

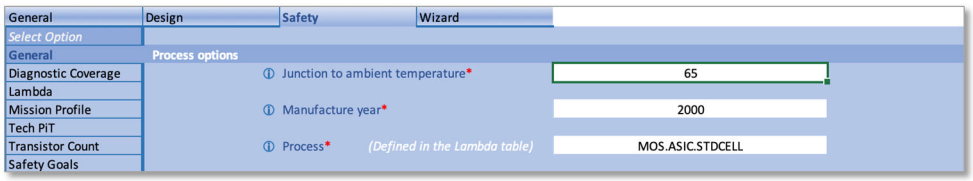


图 8：FIT 计算所需的工艺信息。

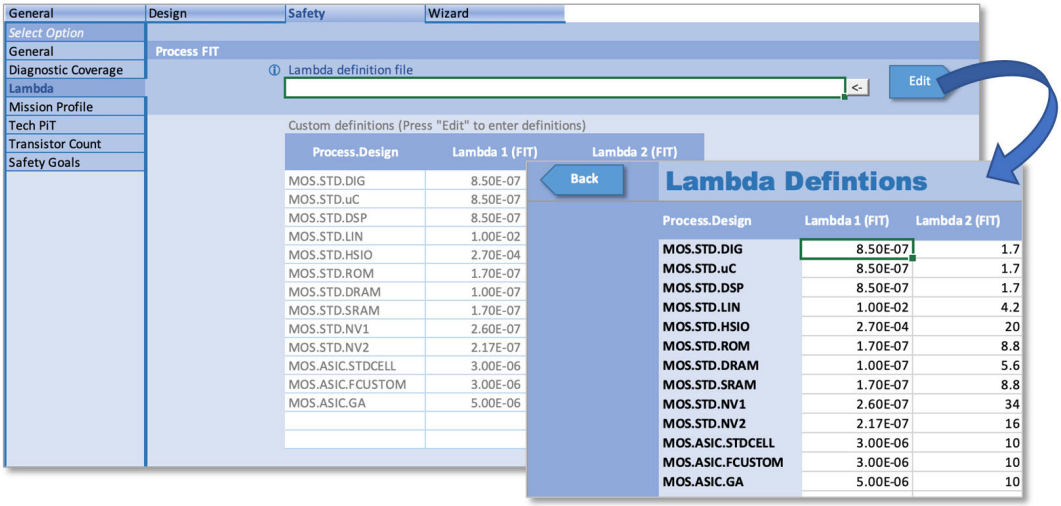


图 9：FIT 计算所需的入信息。

通常，SPFM 和 LFM 的计算是基于单独的安全目标，而不是特定的设计模块或组件。安全目标是设计中最重要的安全要求，所有硬件和软件安全要求都是从其中得出的。我们也在电子表格中定义了这些目标，如图 10 所示，以后可利用这些目标来筛选或汇总每个特定 SPFM 和 LFM 的结果。

图 10：安全目标清单。

创建 FMEDA 的下一步是查看设计层次结构并确定哪些模块与安全相关。这是通过使用仿真设计编译来完成的，我们使用的是 Questasim 工作库（但也可以使用其他工具编译的库）。我们创建了一个向导，以便自动创建 makefile 并提取层次结构（图 11）。

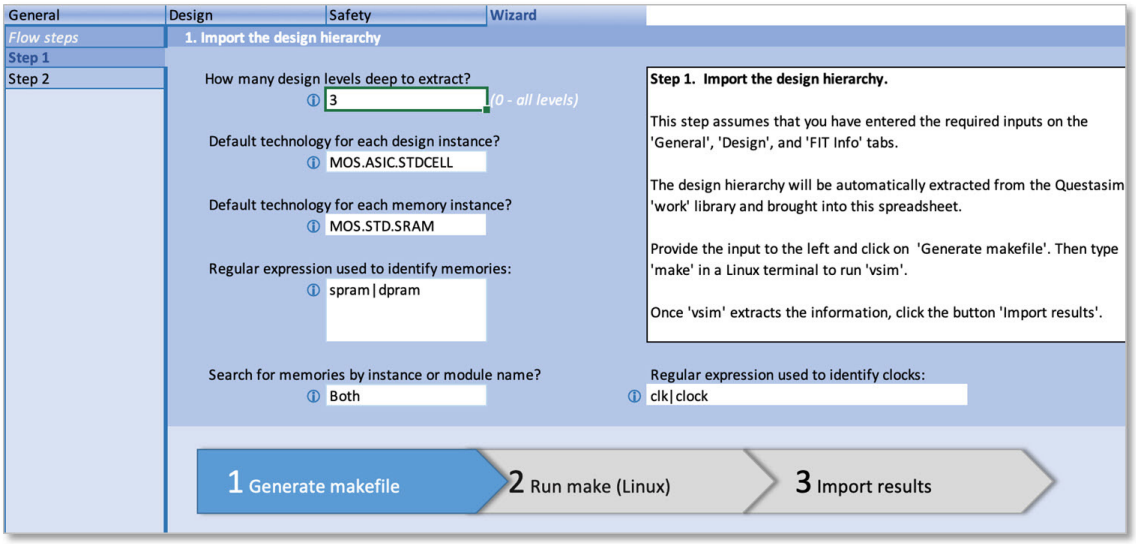
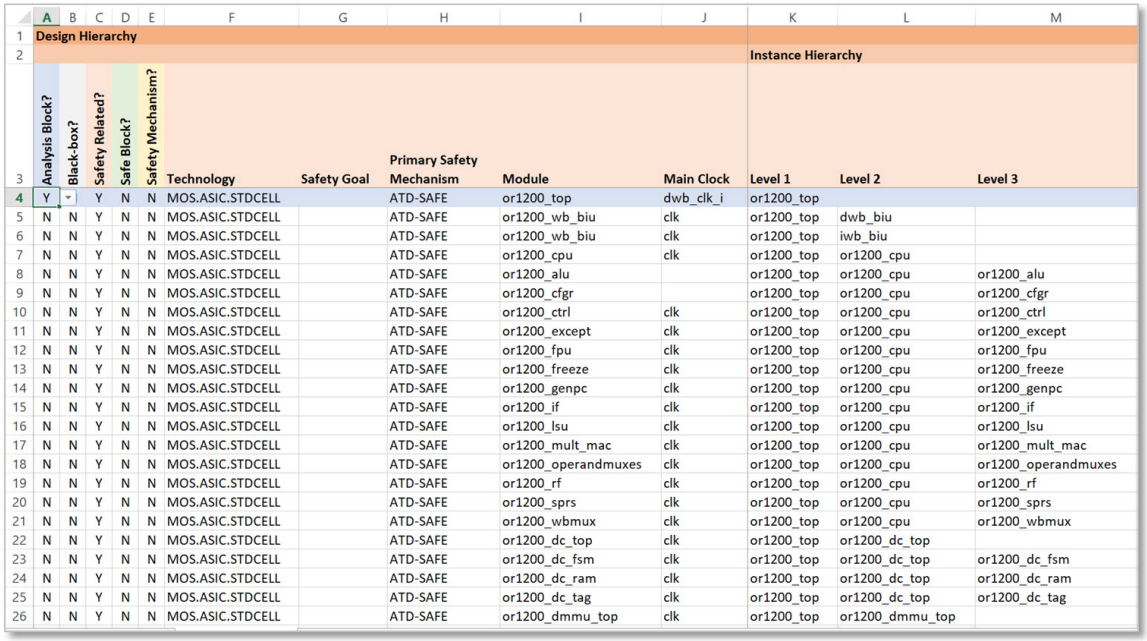


图 11：导入设计层次结构的向导。

过多的层次结构会让 FMEDA 无法管理，因此我们指定设计深度以限制要遍历的层次结构范围。我们还希望分离出存储器（无论其在设计中处于什么级别），因此我们指定一个正则表达式来帮助我们找到设计中的存储器。

有了这些信息，我们便可点击 “Generate makefile”（生成 makefile）来执行宏，从而创建 makefile 和 Tcl 脚本，以将设计和存储器层次结构提取到一个可轻松导入 Excel 的 CSV 文件中。我们切换到 Linux 终端，执行 makefile，然后从仿真器自动提取设计。接下来点击 “Run make (Linux)” 按钮，它提醒我们先运行 Linux makefile，再导入结果。

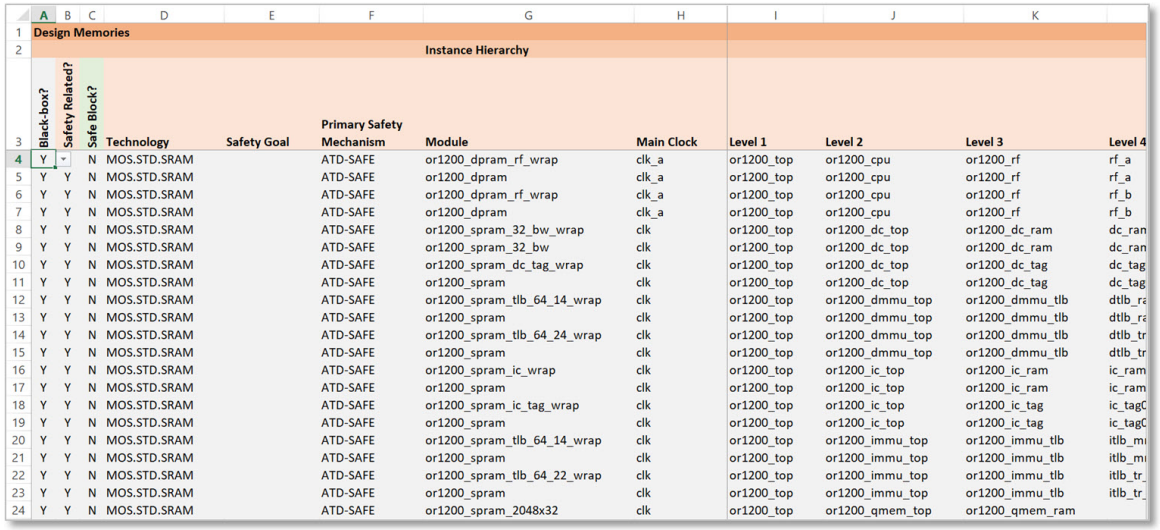
在 Linux 中运行设计提取之后，点击 “Import results”（导入结果），告知 VisualBasic 宏导入并创建设计和存储器层次结构工作表，如图 12 所示。



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Design Hierarchy												
2											Instance Hierarchy		
3	Analysis Block?	Black-box?	Safety Related?	Safe Block?	Safety Mechanism?	Technology	Safety Goal	Primary Safety Mechanism	Module	Main Clock	Level 1	Level 2	Level 3
4	Y	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_top	dwb_clk_i	or1200_top		
5	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_wb_biu	clk	or1200_top	dwb_biu	
6	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_wb_biu	clk	or1200_top	iwb_biu	
7	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_cpu	clk	or1200_top	or1200_cpu	
8	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_alu		or1200_top	or1200_cpu	or1200_alu
9	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_cfgr		or1200_top	or1200_cpu	or1200_cfgr
10	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_ctrl	clk	or1200_top	or1200_cpu	or1200_ctrl
11	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_except	clk	or1200_top	or1200_cpu	or1200_except
12	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_fpu	clk	or1200_top	or1200_cpu	or1200_fpu
13	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_freeze	clk	or1200_top	or1200_cpu	or1200_freeze
14	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_genpc	clk	or1200_top	or1200_cpu	or1200_genpc
15	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_if	clk	or1200_top	or1200_cpu	or1200_if
16	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_lsu	clk	or1200_top	or1200_cpu	or1200_lsu
17	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_mult_mac	clk	or1200_top	or1200_cpu	or1200_mult_mac
18	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_operandmuxes	clk	or1200_top	or1200_cpu	or1200_operandmuxes
19	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_rf	clk	or1200_top	or1200_cpu	or1200_rf
20	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_sprs	clk	or1200_top	or1200_cpu	or1200_sprs
21	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_wbmux	clk	or1200_top	or1200_cpu	or1200_wbmux
22	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_dc_top	clk	or1200_top	or1200_dc_top	
23	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_dc_fsm	clk	or1200_top	or1200_dc_top	or1200_dc_fsm
24	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_dc_ram	clk	or1200_top	or1200_dc_top	or1200_dc_ram
25	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_dc_tag	clk	or1200_top	or1200_dc_top	or1200_dc_tag
26	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_dmmu_top	clk	or1200_top	or1200_dmmu_top	

图 12：自动生成导入的设计层次结构。

此时可将设计模块指定为分析模块，这意味着将单独为其计算 FIT 率。还可以指定其是否与安全相关、是否应将其视为黑盒，以及其是否是一种安全机制。同时输入相应的安全目标和安全机制类型。设计中的存储器被单独拉出到另一个工作表中（图 13）。



	A	B	C	D	E	F	G	H	I	J	K
1	Design Memories										
2							Instance Hierarchy				
3	Black-box?	Safety Related?	Safe Block?	Technology	Safety Goal	Primary Safety Mechanism	Module	Main Clock	Level 1	Level 2	Level 3	Level 4
4	Y	Y	N	MOS.STD.SRAM		ATD-SAFE	or1200_dpram_rf_wrap	clk_a	or1200_top	or1200_cpu	or1200_rf	rf_a
5	Y	Y	N	MOS.STD.SRAM		ATD-SAFE	or1200_dpram	clk_a	or1200_top	or1200_cpu	or1200_rf	rf_a
6	Y	Y	N	MOS.STD.SRAM		ATD-SAFE	or1200_dpram_rf_wrap	clk_a	or1200_top	or1200_cpu	or1200_rf	rf_b
7	Y	Y	N	MOS.STD.SRAM		ATD-SAFE	or1200_dpram	clk_a	or1200_top	or1200_cpu	or1200_rf	rf_b
8	Y	Y	N	MOS.STD.SRAM		ATD-SAFE	or1200_spram_32_bw_wrap	clk	or1200_top	or1200_dc_top	or1200_dc_ram	dc_ran
9	Y	Y	N	MOS.STD.SRAM		ATD-SAFE	or1200_spram_32_bw	clk	or1200_top	or1200_dc_top	or1200_dc_ram	dc_ran
10	Y	Y	N	MOS.STD.SRAM		ATD-SAFE	or1200_spram_dc_tag_wrap	clk	or1200_top	or1200_dc_top	or1200_dc_tag	dc_tag
11	Y	Y	N	MOS.STD.SRAM		ATD-SAFE	or1200_spram	clk	or1200_top	or1200_dc_top	or1200_dc_tag	dc_tag
12	Y	Y	N	MOS.STD.SRAM		ATD-SAFE	or1200_spram_tlb_64_14_wrap	clk	or1200_top	or1200_dmmu_top	or1200_dmmu_tlb	dtlb_r
13	Y	Y	N	MOS.STD.SRAM		ATD-SAFE	or1200_spram	clk	or1200_top	or1200_dmmu_top	or1200_dmmu_tlb	dtlb_r
14	Y	Y	N	MOS.STD.SRAM		ATD-SAFE	or1200_spram_tlb_64_24_wrap	clk	or1200_top	or1200_dmmu_top	or1200_dmmu_tlb	dtlb_tr
15	Y	Y	N	MOS.STD.SRAM		ATD-SAFE	or1200_spram	clk	or1200_top	or1200_dmmu_top	or1200_dmmu_tlb	dtlb_tr
16	Y	Y	N	MOS.STD.SRAM		ATD-SAFE	or1200_spram_ic_wrap	clk	or1200_top	or1200_ic_top	or1200_ic_ram	ic_ram
17	Y	Y	N	MOS.STD.SRAM		ATD-SAFE	or1200_spram	clk	or1200_top	or1200_ic_top	or1200_ic_ram	ic_ram
18	Y	Y	N	MOS.STD.SRAM		ATD-SAFE	or1200_spram_ic_tag_wrap	clk	or1200_top	or1200_ic_top	or1200_ic_tag	ic_tagC
19	Y	Y	N	MOS.STD.SRAM		ATD-SAFE	or1200_spram	clk	or1200_top	or1200_ic_top	or1200_ic_tag	ic_tagC
20	Y	Y	N	MOS.STD.SRAM		ATD-SAFE	or1200_spram_tlb_64_14_wrap	clk	or1200_top	or1200_immu_top	or1200_immu_tlb	itlb_m
21	Y	Y	N	MOS.STD.SRAM		ATD-SAFE	or1200_spram	clk	or1200_top	or1200_immu_top	or1200_immu_tlb	itlb_m
22	Y	Y	N	MOS.STD.SRAM		ATD-SAFE	or1200_spram_tlb_64_22_wrap	clk	or1200_top	or1200_immu_top	or1200_immu_tlb	itlb_tr
23	Y	Y	N	MOS.STD.SRAM		ATD-SAFE	or1200_spram	clk	or1200_top	or1200_immu_top	or1200_immu_tlb	itlb_tr
24	Y	Y	N	MOS.STD.SRAM		ATD-SAFE	or1200_spram_2048x32	clk	or1200_top	or1200_qmem_top	or1200_qmem_ram	

图 13：对模块 / 实例名称使用正则表达式从设计中自动提取的设计中的存储器。

Excel 支持基于条件的高亮显示，因此设计或存储器工作表更易于查看和管理（图 14）。

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Design Hierarchy												
2											Instance Hierarchy		
3	Analysis Block?	Black-box?	Safety Related?	Safe Block?	Safety Mechanism?	Technology	Safety Goal	Primary Safety Mechanism	Module	Main Clock	Level 1	Level 2	Level 3
18	Y		Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_operandmuxes	clk	or1200_top	or1200_cpu	or1200_operandmuxes
19	Y	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_rf	clk	or1200_top	or1200_cpu	or1200_rf
20	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_sprs	clk	or1200_top	or1200_cpu	or1200_sprs
21	N	N	N	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_wbmux	clk	or1200_top	or1200_cpu	or1200_wbmux
22	Y	N	Y	N	N	MOS.ASIC.STDCELL	1	ATD-SAFE	or1200_dc_top	clk	or1200_top	or1200_dc_top	
23	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_dc_fsm	clk	or1200_top	or1200_dc_top	or1200_dc_fsm
24	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_dc_ram	clk	or1200_top	or1200_dc_top	or1200_dc_ram
25	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_dc_tag	clk	or1200_top	or1200_dc_top	or1200_dc_tag
26	Y	N	Y	N	N	MOS.ASIC.STDCELL	2	ATD-E2EECC	or1200_dmmu_top	clk	or1200_top	or1200_dmmu_top	
27	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_dmmu_tlb	clk	or1200_top	or1200_dmmu_top	or1200_dmmu_tlb
28	N	N	Y	N	Y	MOS.ASIC.STDCELL		ATD-SAFE	or1200_du	clk	or1200_top	or1200_du	
29	Y	N	Y	N	N	MOS.ASIC.STDCELL	1	ATD-SAFE	or1200_ic_top	clk	or1200_top	or1200_ic_top	
30	N	N	Y	N	N	MOS.ASIC.STDCELL		ATD-SAFE	or1200_ic_fsm	clk	or1200_top	or1200_ic_top	or1200_ic_fsm

图 14：内置基于条件的高亮显示以提高可读性。

设计层次结构工作表使得在安全分析中添加和删除组件变得非常容易，这对于假设分析非常有用。FMEDA 生成是自动进行的，您可以循环访问不同的安全机制，重新生成 FMEDA 和 FIT 率，并观察其对诊断覆盖率和整体 FIT 计算的影响。

下一步是实际计算 FIT 和诊断覆盖率（图 15）。同样使用 VisualBasic 来创建所有 makefile、输入和输出目录以及用于 FIT 分析的输入文件（图 16）。与导入设计层次结构一样，有一个步骤需要在 Linux 中完成，即为每个分析模块运行 FIT 计算。较低级别模块上的 FIT 计算必须首先执行，以便可以将 FIT 结果用于更高设计层次结构中。使用生成的 makefile 自动处理所有依赖关系。

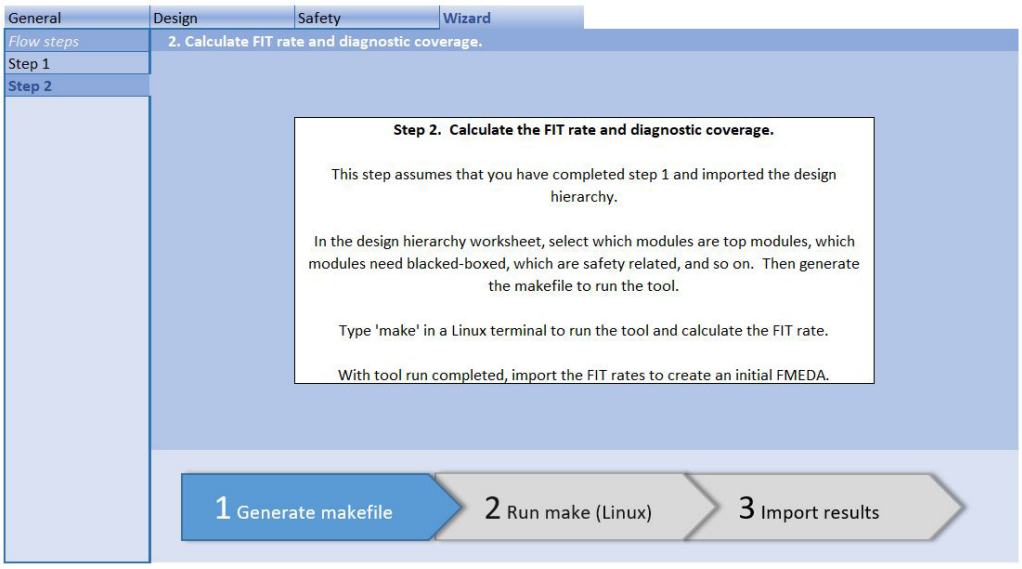


图 15：第 2 步运行 FIT 计算并将结果导入 FMEDA。

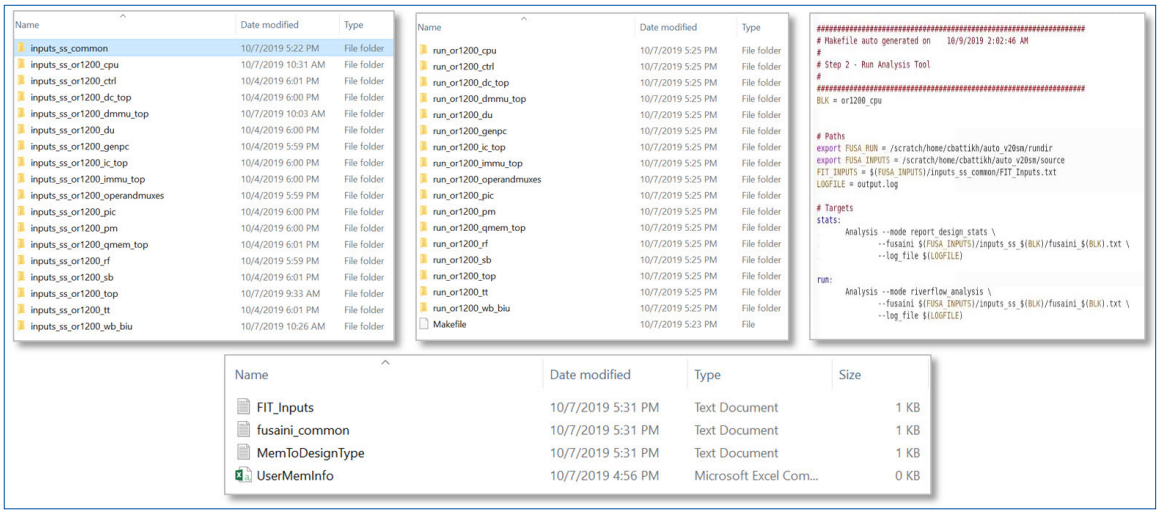


图 16：向导创建的目录和文件，用以生成 FIT 计算。

完成 FIT 和诊断覆盖率的计算之后，点击 “Import results”（导入结果）按钮以加载每个分析过的模块和存储器的 FIT 计算值，然后将其放入自动生成的 FMEDA 中（图 17）。

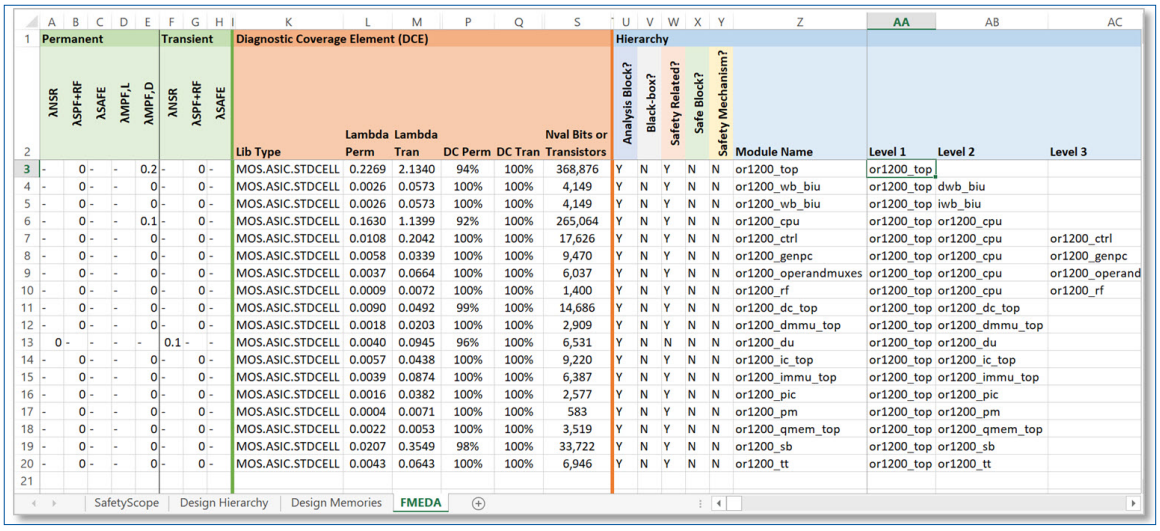


图 17：使用 FIT 计算自动生成的 FMEDA。

由此得到的 FMEDA 可用作项目的 FMEDA，或用于假设分析安全探索。由于它是自动生成的，因此可以在 “设计层次结构” 工作表中尝试不同的安全机制，FMEDA 会非常快速地重新生成。如果您使用安全机制插入工具（通过插入奇偶校验、ECC、冗余、表决等保护措施来自动增强设计安全性）的话，上述尝试可能特别有用。

下一步是什么？

文中我们讨论了仅在安全机制的模块级别使用该解决方案。然而，实际的 IC 设计可能有跨多个模块的安全机制，尤其是当寄存器分布在模块边界时。另外，安全机制可能仅占用一个模块的一部分，或从分散在多个模块中的单个寄存器中提取信息。通过指定安全机制的影响锥，而不只是周围模块，可以进行更详细、更准确的分析 [8]。

当然，只有使用实际失效活动注入失效并对结果进行分类，才能获得真正的诊断覆盖率 [3]。失效活动之后，将实际诊断覆盖率插回到 FMEDA 中。使用循环失效仿真器以自动化方式运行，有望能够同时运行数千个失效注入仿真。如本文所述，EDA 工具可以理解和读取结果。同样，硬件加速仿真是测试软件安全机制的理想选择。Mentor 的硬件加速器 Veloce 有一个 FaultApp，它能在注入失效的同时运行基于软件的安全机制，例如循环冗余校验 (CRC)。这些结果连同使用 SafeCheck（基于 Questa Formal 构建的 Mentor 咨询流程）进行形式验证分析所得到的结果，可以写成脚本合并回 EDA 工具中。在失效活动中完成对所有失效的分类之后，便可计算最终的诊断覆盖率指标。

利用本文介绍的工具和方法让繁琐的 FMEDA 创建任务实现自动化，您将有更多时间来专心探索设计的安全就绪情况，并弄清楚如何才能更好地增强设计安全性。如果您在启动、自动化或执行功能安全过程方面需要帮助，Mentor 咨询服务可随时帮助您将您的汽车设计成功推向市场。有关我们的使命以及如何联系我们的更多信息，请访问：
https://www.mentor.com/training-and-services/consulting-services

参考文献和其他阅读内容

1 ISO 26262:2018. Road vehicles — Functional Safety. International Standards Organization.

2 How to Stay Out of the News with ISO 26262-Compliant Verification, Tutorial Session. DVCon, San Jose CA, 1 March 2018.

3 It's Not My Fault! How to Run a Better Fault Campaign Using Formal, White Paper, Mentor Graphics, October 2019.

4 Formal Flow for Automotive Safety: Bulletproofing Car Design, Circuit Cellar Magazine, Issue 344, March 2019.

5 Win on the Fault Campaign Trail with Formal, Tech Design Forum, Dec. 11, 2018.

6 Formal Fault Analysis for ISO 26262: Find Faults before They Find You, Tech Design Forum, June 18, 2018.

7 How Formal Concentrates ISO 26262 Fault Analysis, Tech Design Forum, March 2, 2017.

8 How Formal Reduces Fault Analysis for ISO 26262, White Paper, Mentor Graphics, January 2017.

如需最新信息，请致电联系我们，或者访问：

www.mentor.com

公司总部
Mentor Graphics Corporation
8005 S.W. Boeckman Road
Wilsonville, Oregon 97070 USA
电话：+1-503-685-7000
传真：+1-503-685-1204

销售和产品信息
电话：+86-21-6101-6301
sales_info@mentor.com

上海
明导（上海）电子科技有限公司
上海市浦东新区杨高南路 759 号
陆家嘴世纪金融广场 2 号楼 5 楼
邮编：200127
电话：+86-21-6101-6301
传真：+86-21-5047-1379

北京
明导（上海）电子科技有限公司
北京办事处
北京市南礼士路 66 号
建威大厦 1512 室
邮编：100045
电话：+86-10-5930-4001
传真：+86-10-6808-0319

深圳
明导（上海）电子科技有限公司
深圳办事处
深圳市福田区金田路 3088 号
中洲大厦 24 楼 2401 室
邮编：518040
电话：+86-755-8282-2700
传真：+86-755-8826-7750

Mentor® A Siemens Business

MGC 06-20 TECH18600-w-CN