2014年 12月 18日 (木)

仕向け切り替えのための自動生成コードカスタマイズ

本日の内容

1. はじめに：自動生成コードのコード効率について
2. Variantサブシステムについて
3. Variantサブシステムからのコード生成
4. 最適なROM/RAM消費実現のための
   カスタマイゼーション
5. まとめ(効果と制限)

株式会社デンソー

電子基盤システム開発部
モデルベース技術開発室

福田 勝吉

DENSO

©DENSO CORPORATION All rights reserved.
This information is the exclusive property of DENSO CORPORATION. Without their consent, it may not be reproduced or given to third parties.

1. はじめに: 自動生成コードのコード効率について

弊社の使い方に置けるコード効率(ROM/RAM容量)の改善傾向

R2006* R2007* R2008* R2009* R2010* R2011* R2012* R2013* R2014*

RAM

BlockI/O構造体の削減による改善継続

ROM

R2010b以降、コード展開の改良あるもROM容量に変化なし

傾向は、JMMAB / JMAB等で、NDAのもと、MathWorksから開示されている情報と矛盾しない

弊社の使い方でコード効率が悪化する要因

1. ポインタ → 我慢

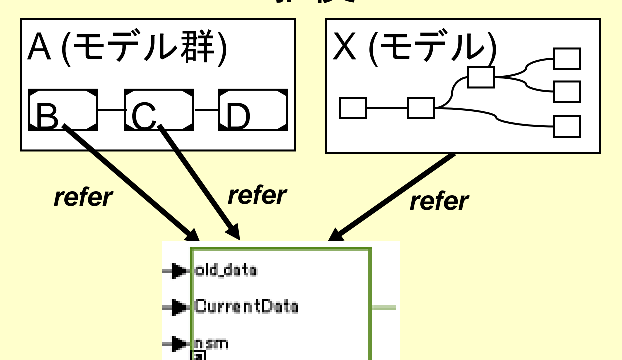
<ハンドコード>

```
if (***) { ptt_map = &Amap[0];
} else {   ptt_map = &Bmap[0]; }
out = LookUp(in,  ptt_map);
```

<オートコード>

```
if (***) {out = LookUp(in, &Amap[0];);
} else {   out = LookUp(in, &Bmap[0];); }
```

2. モデルを跨いだ使い回し → reusable駆使



3. 複雑な仕向け切替

→ Variant Subsystemだけではプロジェクト運用に支障

→ 最低限必要な機能を、カスタマイズにより実現

(本日の説明)

© DENSO CORPORATION All rights reserved.
This information is the exclusive property of DENSO CORPORATION. Without their consent, it may not be reproduced or given to third parties.

2-1. Variantサブシステムについて(1/2)

使い方

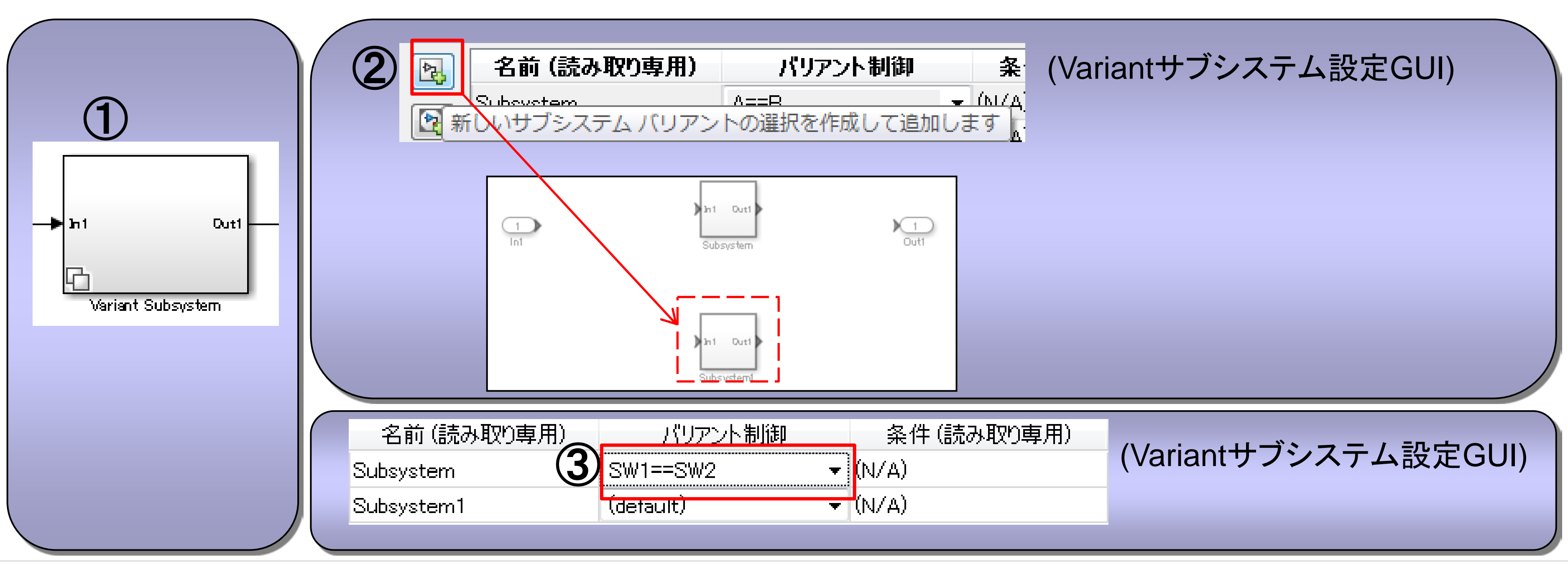
①VariantサブシステムをSimulinkライブラリからコピー

②Variant下に仕向け分のサブシステムを作成

③Variantに仕向け条件式を設定(右クリックでGUI起動)

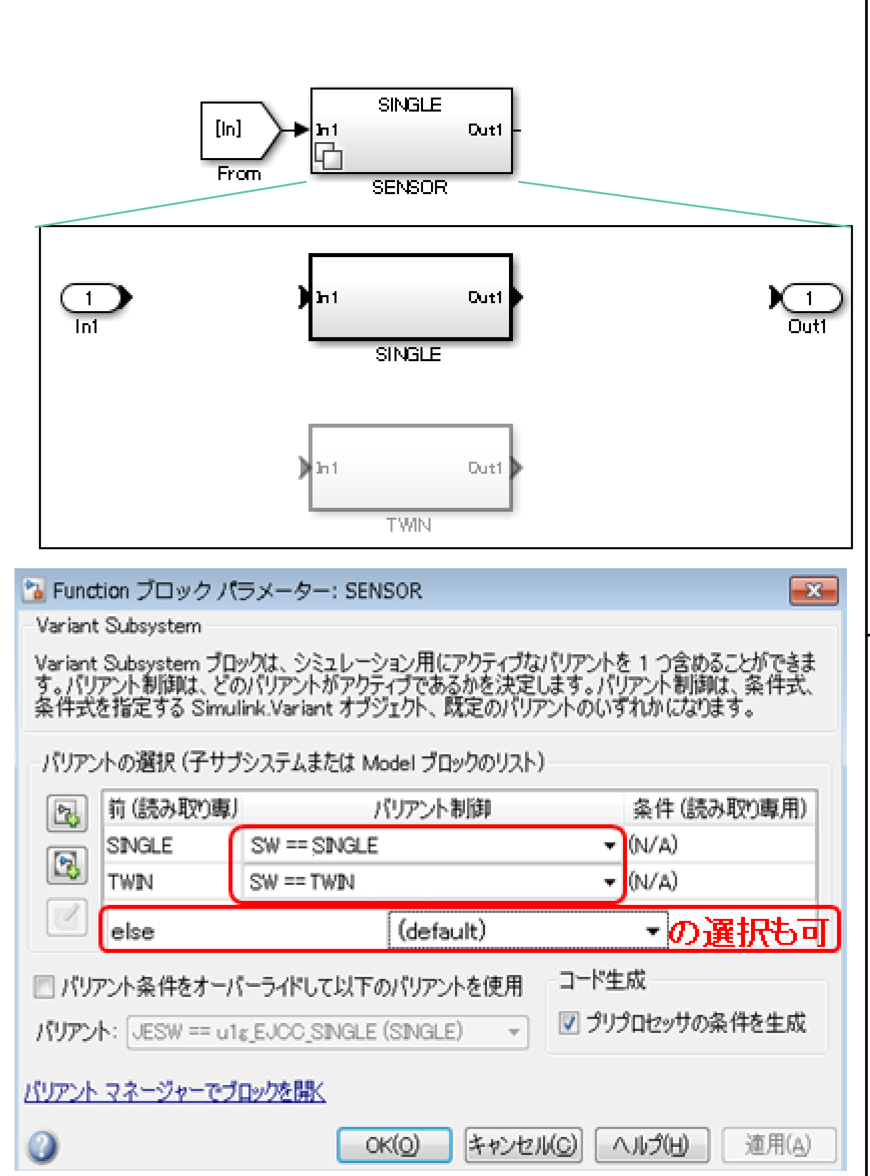
④WSに条件式に利用された定数を定義

例：- SW1 = Simulink.Paramter; SW1.Value = 1;



© DENSO CORPORATION All rights reserved.
This information is the exclusive property of DENSO CORPORATION. Without their consent, it may not be reproduced or given to third parties.

2-2. Variantサブシステムについて(2/2)



インライン展開や#elifの生成が可能

```
#if SW == SINGLE
        s2t_Sw1 = variantsample_B.USE_b.iram[0];
#elif SW == TWIN
        s2t_Sw1 = ((variantsample_B.USE_b.iram[0]) +
                   (variantsample_B.USE_b.iram[1]));
#endif
```

関数(サブシステム)の引数渡しも可能

```
#if SW == SINGLE
        variantsample_SINGLE(variantsample_B.USE_b.iram,
                             &s2t_Sw1);
#elif SW == TWIN
        variantsample_TWIN(variantsample_B.USE_b.iram,
                           &s2t_Sw1);
#endif
```

その他：Variantマネージャによる設計補助機能

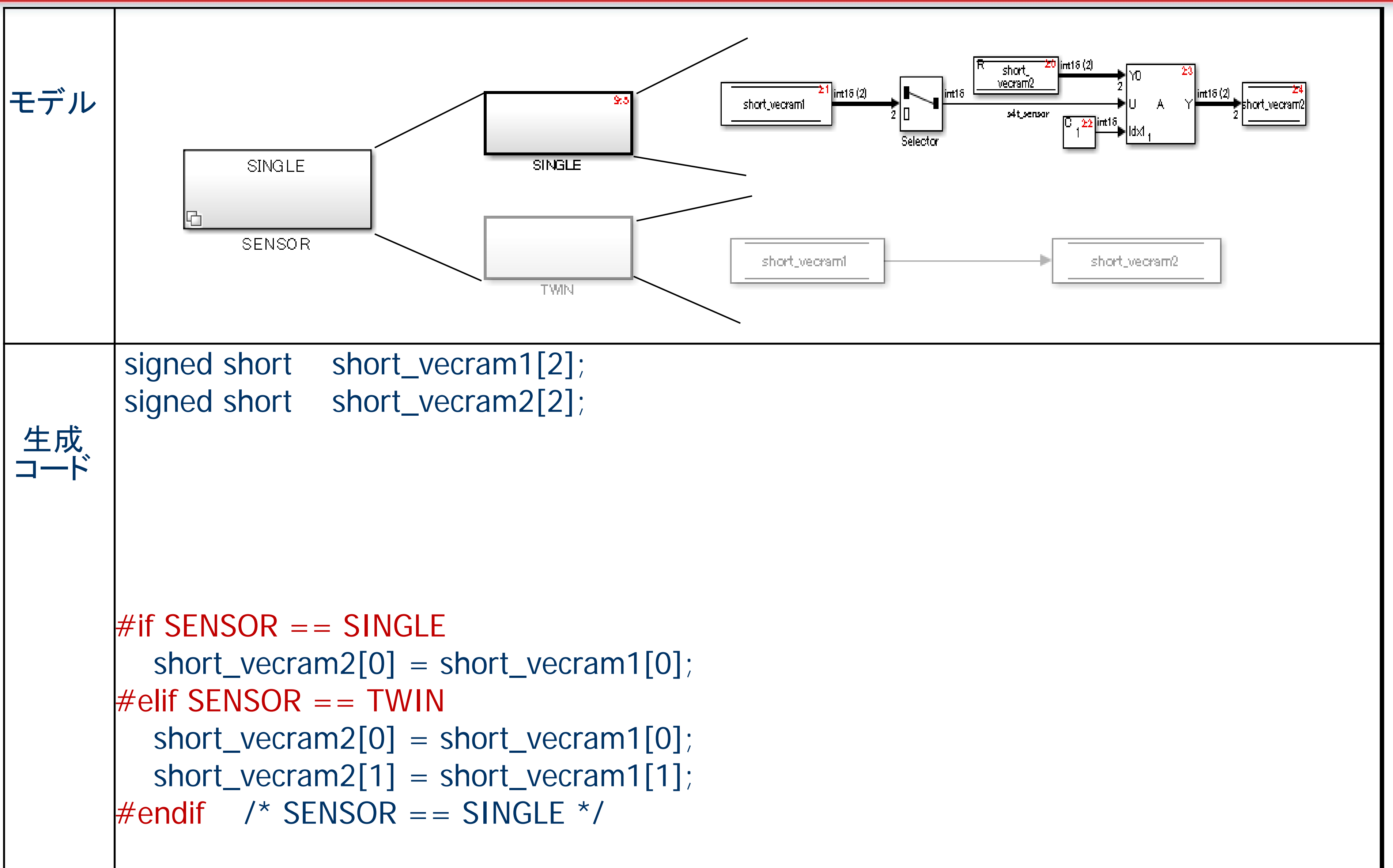
- モデル全体のコンパイルSWの可視化
- 変数・コンフィギュレーションの管理
- 矛盾のチェック

バリアント マネージャー: variantsample_R2014bOK
バリアント コンフィギュレーション データ
名前
コンフィギュレーション
制約
コンフィギュレーション:
制御変数
サブモデル コンフィギュレーション
サブモデル
コンフィギュレーション
モデルの階層構造 (ベース ワークスペースを使用)
検証 (ベース ワークスペース...
表示 Model ブロックとバリ...

名前	バリアント制御
variantsample_R2014bOK	
template	
template_8msh1_Flp	
VARI1	
USE	VARI1 == USE
SENSOR	
SINGLE	SENSOR==SINGLE
TWIN	SENSOR==TWIN
type5	
Subsystem	
Subsystem	VARI2 == NOT_USE

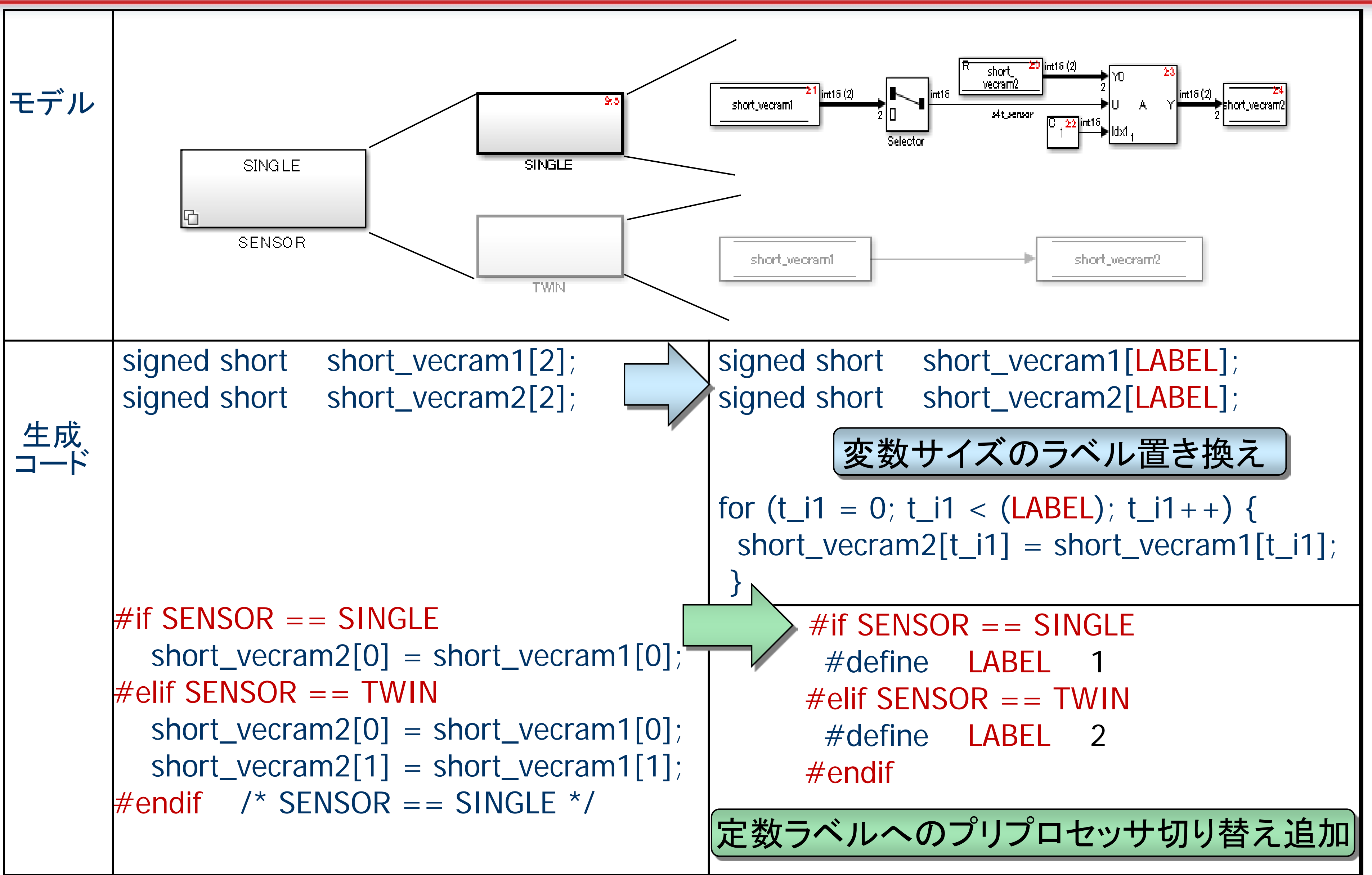
©DENSO CORPORATION All rights reserved.
This information is the exclusive property of DENSO CORPORATION. Without their consent, it may not be reproduced or given to third parties.

3-1. Variantサブシステムからのコード生成

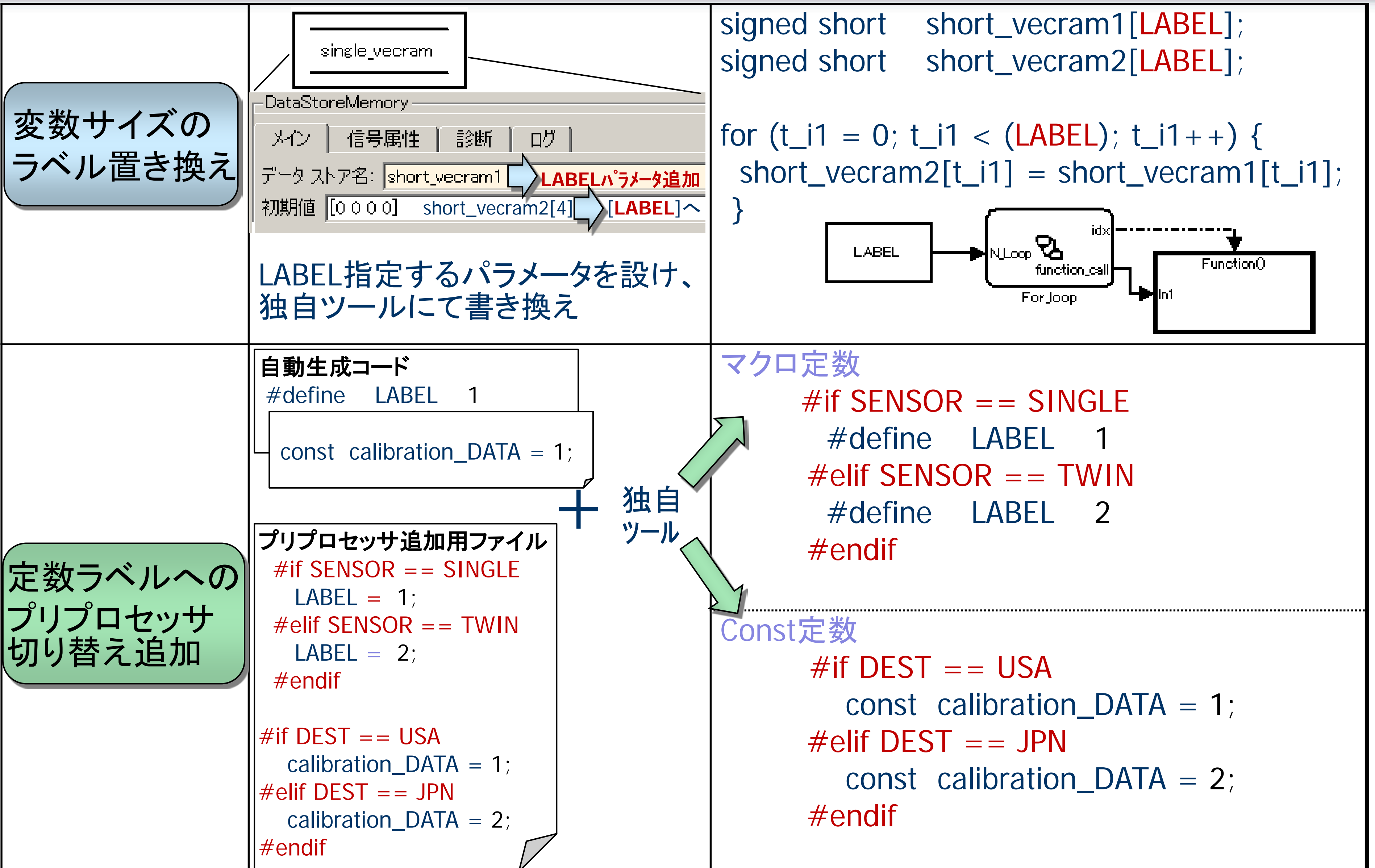


© DENSO CORPORATION All rights reserved.
This information is the exclusive property of DENSO CORPORATION. Without their consent, it may not be reproduced or given to third parties.

3-2. Variantサブシステムからのコード生成(ユーザ要望)



© DENSO CORPORATION All rights reserved.
This information is the exclusive property of DENSO CORPORATION. Without their consent, it may not be reproduced or given to third parties.



© DENSO CORPORATION All rights reserved.
This information is the exclusive property of DENSO CORPORATION. Without their consent, it may not be reproduced or given to third parties.

5. まとめ(効果と制限)

	効果	制限(個別アプローチの限界)

変数サイズのラベル置き換え	1. 変数サイズの動的確保
	(常に最大値を確保する運用に比べRAMの効率利用)
	short_vecram1[LABEL];
	
	2. 配列違いのコードを同一ロジックにまとめられる事による検査効率の向上
	for (t_i1 = 0; t_i1 < (LABEL); t_i1++)
	{short_vecram2[t_i1] }
		関数引数のベクトルの扱い
		関数コール側はOKだが.
		Reusablefunc(vecram);
		Enablefunc(en,vecram);
		
		関数実体ではBlockI/Oに、
		サイズ情報が含まれNG
		void func(const f4 rtu_In1[4])
		{
		vecram[0] = rtu_In1[0];
		・・・
		vecram[0] = rtu_In1[4];
		
		short_vecram / 4 / In1 / ReusableSubsystem
		short_vecram / 4 / In1 / EnableSubsystem
		
		このような機能が将来、標準サポートされる事を希望

定数ラベルへのプリプロセッサ切り替え追加	3. 同一のConstant定数名の複数仕向け適用
	(仕向毎にラベル定義するのに比べROMの効率利用)
	#if DEST == USA
	tempram =calibration_USA;
	#elif DEST == JPN
	tempram =calibration_JPN;
	
	4. 従来ハンドコードからの同一ラベル名での移植
		<プリプロセッサ追加用ファイル>
		#if DEST == USA
		DATA1 = 1;
		#elif DEST == JPN
		DATA1 = 2;
		#endif
		
		#if DEST == USA
		DATA2 = DATA1; ←ファイル内で定義済みのラベル参照
		#elif (DEST != USA) && (DEST == JPN) && (DEST == EUR) ←整合性
		DATA1 = DATA2; ←矛盾や間違い
		#endif
		
		※生成したコードのModifyでは、ユーザーの複雑な設定への対応に限界.(モデル情報を用いたチェックが必要)

© DENSO CORPORATION All rights reserved.
This information is the exclusive property of DENSO CORPORATION. Without their consent, it may not be reproduced or given to third parties.