ヘテロジニアスマルチプロセッサ向けの通信機構の評価

2019SC002 江塚俊介
指導教員：本田晋也

1　はじめに

高性能が要求されるカメラや TV 等の組込みシステムではマルチプロセッサの利用が広がっている．同じ命令セットの複数のプロセッサを 1 つの OS により制御するホモジニアスマルチプロセッサがこれまで一般的であった．

一方，処理能力のさらなる向上のために複数の異なるプロセッサを混在させるヘテロジニアスマルチプロセッサも増えている [1]. 例として本研究で対象とする i.MX RT や STM32H7 等が挙げられる．AI スピーカーはユーザからのウェイクアップコマンドを常に待つ必要があり，消費電力が少ないサブのプロセッサを常に稼働させることで消費電力を抑える．制御システムでは計算処理と IO 処理を別のプロセッサで実現する場合，計算処理は高性能の規模の大きなプロセッサを使用し，IO 処理は割り込み応答性の高い小規模なプロセッサを使用する．

ヘテロジニアスマルチプロセッサはプロセッサ毎に命令セットが異なるため，各プロセッサで異なる OS を用いる必要がある．異なる OS を用いるとプロセッサ間の通信を OS とは別のソフトウェアモジュールで実現する必要がある．ヘテロジニアスマルチプロセッサ向けの OS 間通信機構としては Remote Processor Massaging（RPMsg）と呼ばれる仕様が広く用いられている．RPMsg は共有メモリと割り込みを組み合わせて OS 間の通信を実現している．RPMsg は標準的な仕様が存在するのではなく，ヘテロジニアスマルチプロセッサを作成している各チップベンダが独自に実装を公開している．そのためベンダ間の互換性が低く，仕様に関しても明確なドキュメントが存在しない．メインプロセッサの OS として Linux を対象とした RPMsg の性能評価はされているが [2][3]，RTOS を対象とした RPMsg の通信評価はされていない．

本研究では，組込みシステムで広く使われている ARM プロセッサと，AI や音声処理を高速に処理するカスタム命令を持つ Digital Signal Processor（DSP）である Xtensa HiFi4 のヘテロジニアスマルチプロセッサを対象に RPMsg の解析と性能評価，RAM の使用量削減について試行する．

2　i.MX RT685

本研究で対象とする NXP 社のヘテロジニアスマルチプロセッサである i.MX RT685 について説明する．i.MX RT685 の構成図を図 1 に示す．プロセッサとしては，ARM と DSP を搭載している．各プロセッサが使用するデータは，チップ内部の SRAM に配置する．SRAM は複数バンクあり，ARM と DSP の両方からアクセス可能である．各プロセッサが使用するプログラムは，チップ外部のフラッシュメモリに配置する．フラッシュメモリはチップ外であるため，高速化のためにキャッシュメモリを使用する．加えて DSP は高速化のため Tightly-CoupledMemory（TCM）と呼ばれるローカルメモリにプログラムを配置可能である．

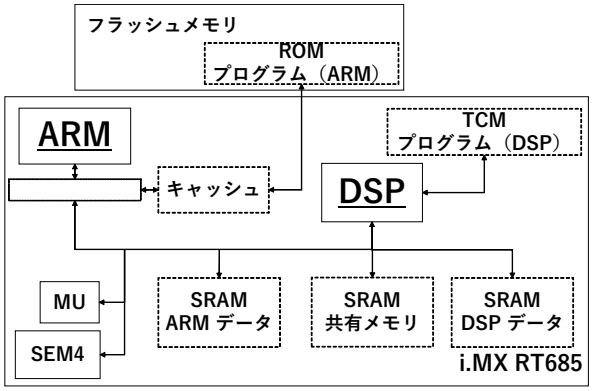

図 1　i.MX RT685 の構成図

3　RPMsg の解析

RPMsg のコードを解析し仕様と実装を明らかにした．

RPMsg はエンドポイントオブジェクト，キューオブジェクト，バッファを使用して通信する．エンドポイントオブジェクトは属性にアドレス，受信時コールバック関数を持つ．

ARM と DSP の送受信のシーケンスを図 2 に示す．ARM 側は OS あり，DSP 側は OS なしの環境である．

送信は，rpmsg_send() に送信するデータのポインタ，送信元のエンドポイント（ept）と送信先のエンドポイントのアドレス（dst）を引数に指定することで行う．

受信は OS ありとなしで実現方法が分かれており，OS ありの場合は，rpmsg_queue_recv() を呼ぶ．その先では，OS のキュー機能を使用して，キューからデータを受信する xQueueReceive() を呼び出し，データが格納されるまで受信待ち状態になる．データが送信されると，RPMsg 内のコールバック関数から xQueueSend() が呼び出されキューにデータが格納され，xQueueReceive() による待ちが解除されてデータを受信する．OS なしの場合はコールバック関数が呼び出される（図 2 中の DSP 受信）．

4　RPMsg の性能評価

RPMsg の基本的な性能を評価するため，図 2 に示したシーケンスを計測する．また RPMsg の通信速度の妥当性を考察するために ARM のタスク間通信にかかる時間を計測する．

1. **ARM-DSP 送信時間**：ARM がデータを送信してから DSP がデータを受信するまでの時間（図 2 の 1）

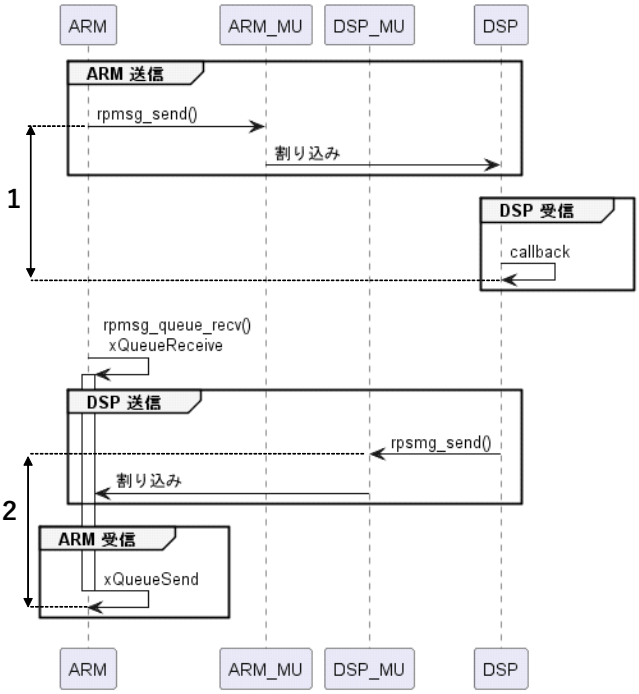

図 2　送受信シーケンス

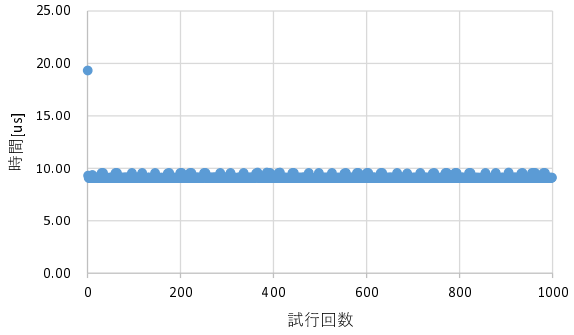

図 3　ARM-DSP 送信時間

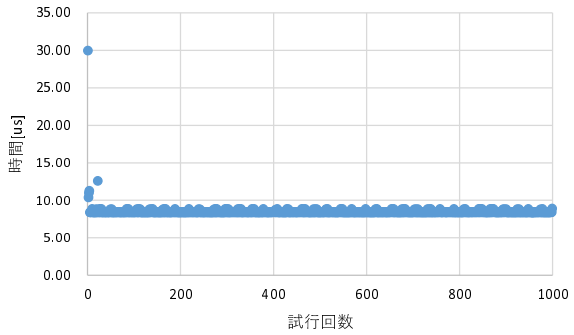

図 4　DSP-ARM 送信時間

2. **DSP-ARM 送信時間**：DSP がデータを送信してから ARM がデータを受信するまでの時間（図 2 の 2）

計測は，シーケンスの各ポイントを実行すると GPIO の状態を変化させ，その変化をロジックアナライザーを用いて計測した．なお，ARM の動作周波数は 300Mhz，DSP の動作周波数は 600Mhz である．

計測結果を図 3，図 4 に示す．結果より，RPMsg の通信時間はおおよそ一定である．試行回数の 1 回目が他の時間と比較して遅いのは，ARM 側のプログラムは図 1 のチップにはなく ROM にあり，1 回目はキャッシュにデータがないため，データの取り出しに時間がかかるからである．

ARM において，タスク間のキューで同じサイズのデータを送受信した場合の実行時間（コア内通信の実行時間）は 4.7us であったため，2 倍程度の時間でプロセッサ間通信が可能であることが分かった．

5　RAM 使用量削減

RAM は ROM とサイズを比較して一桁小さいため，RAM の使用量を減らしたいという要求がある．これらの問題点を解決する方法として ROM 化がある．ROM 化とはソフトウェアで用いられるデータ構造のうち，動的に確保する必要がなく，初期化時に一度値を設定すると書き換える必要がないものを ROM データとする方法である．

今回は，エンドポイント毎に必要なデータ構造（エンドポイント管理構造体）を対象に ROM 化の試行を行った．RPMsg には，オプションとして，エンドポイント管理構造体を動的メモリ確保でなく，静的にメモリ確保するオプションが用意されている．この機能を有効にした上で，このデータのうち，初期化時以外に書き換えられることのない変数を調査し，それらの変数を const 型の変数として，宣言時に初期値を設定することで ROM データとした．この変更は，RPMsg のプログラムを 5 箇所変更することで実現することができた．結果として，エンドポイント 1 つあたり 12 バイトの RAM の使用量が削減できた．

6　おわりに

本研究では RPMsg の解析，性能評価，RAM 使用量削減を行った．性能評価では RPMsg の性能評価を行い OS を用いたコア内通信の 2 倍の時間であることを示した．RAM 使用量削減では ROM 化の試行を行い，エンドポイント 1 つあたり 12 バイトの RAM 使用量削減に成功した．

参考文献

[1] Niu Bin, Li Dejian, Bai Zhihua, He Longlong, Zhang Guang and Li Meng : *Asymmetric software architecture design of High performance control chip applied in industrial control field*, 2021 4th International Conference on Advanced Electronic Materials, Computers and Software Engineering(AEMCSE), pp.26-28 (2021).

[2] Sara Alonso, Jesus Lazaro, Jaime Jimenez, Leire Muguira and Unai Bidarte : *Evaluating the OpenAMP framework in real-time embedded SoC platforms*, 2021 XXXVI Conference on Design of Circuits and Integrated Systems(DCIS), pp.24-26 (2021).

[3] 大竹史紘, 本田晋也, 高田広章：ヘテロジニアスプロセッサ向け通信ライブラリ MDCOM，研究報告組込みシステム (EMB)， vol.34，pp.1-6 (2017).