渡部研究室

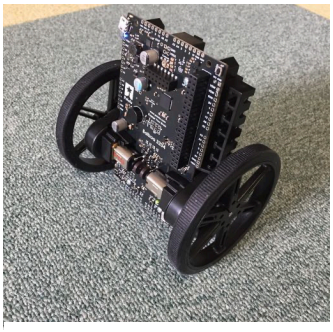
Programming Systems Group

情報理工学院・情報工学系（www.psg.c.titech.ac.jp）
渡部卓雄(教授)・森口草介(助教)

渡部研究室では，マイクロコントローラ等の計算資源が限定された組込みシステムを対象とする，関数リアクティブプログラミング(FRP)言語に関する研究を行っています．これまでに，オリジナルのFRP言語Emfrp等の開発を通して，小規模組込みシステム開発におけるFRPの有用性を示してきました．現在，FRP言語を対象に，データサイズを静的に規定可能な型システム，非同期計算の導入，状態遷移モデルの導入，並列・分散実行方式，実時間性の表現，ハイブリッドアーキテクチャにおける実装方式，形式化と検証，および各種の応用について研究を行なっています．

リアクティブシステムとFRP

組込みシステムは，センサやスイッチ，通信ポートを介した外界からの入力に反応し，モータやディスプレイ等の機器を制御します．このような，外界からの入力に反応し続けるような動作をするシステムを**リアクティブシステム**と呼びます．組込みシステムの他にも，GUI，Webアプリケーション，ゲーム等はリアクティブシステムとみなすことができます．
一般にリアクティブシステムに与えられる入力は順序やタイミングが予測できません．そのため，ポーリングや割り込み，イベントシステム，状態機械と呼ばれる手法がリアクティブシステムのプログラミングにおいて現在一般的に用いられています．ところが，これらの手法を用いるとプログラムが分断されて非常に見通しが悪くなり，作成や保守のコストが増大します．
関数リアクティブプログラミング(FRP)は，副作用のない式のみを用いてリアクティブシステムのプログラムを記述しようという考え方です．これによってプログラムの見通しをよくし，保守コストを下げようという目的で研究が行われています．



倒立振子ロボット Pololu Balboa 32U4 ATmega 32U4 (32KB Flash, 2.5KB RAM)

Emfrp

Emfrp[1]は渡部研究室で開発された小規模組込みシステム向け関数リアクティブプログラミング(FRP)言語で，組込みシステムのさまざまな動作を簡潔に記述することができます．FRPの中心的なアイディアは時々刻々と変化するデータをそのまま表現する時変値(Time Varying Value)と呼ばれる言語機構です．例えば天井と床に設置された温度センサの値とファンのON/OFF状態を表す時変値をそれぞれ tceil, tfloor, fan とすると，温度差が5度を超えている間にファンを回すプログラムは以下のようにわずか1行で書くことができます．

node fan : Bool = abs(tceil - tfloor) > 5

さらにEmfrpでは，プログラムが使用するメモリの大きさがコンパイル時に決定されることと，時変値の更新処理（nodeで始まる文における=の右辺の計算）は必ず停止することが保証されています．したがって，Emfrpはマイクロコントローラのような計算資源が限定されたシステムで動作するプログラムの開発に適しています．

```
module Balancer
in  gyroY  : Int,  # gyroscope (y-axis)
    accX   : Int,  # accelerometer (x-axis)
    encL   : Int,  # left motor encoder
    encR   : Int   # right motor encoder
out motorL : Int,  # left motor
    motorR : Int,  # right motor
use StdL

...

node init[0] angle = (angle@last + gyroY * update_time_ms) * 99 / 100

node init[(0, 0)] (encL1, encR1) = (encL, encR)
node (speedL, speedR) = (encL1 - encL1@last, encR1 - encR1@last)
node init[(0, 0)] (distL, distR) = (distL@last + speedL, distR@last + speedR)

node risingAngleOff  = gyroY * angle_rate_ratio + angle
node fallingAngleOff = gyroY * angle_rate_ratio - angle

node init[0] motor =
    limSpeed(motor@last +
             (angle_resp * risingAngleOff +
              dist_resp * (distL + distR ) +
              speed_resp * (speedL + speedR)) / 100 / gear_ratio)

node (motorL, motorR) =
    if accX > 0 then (motor + (distL - distR) * dist_diff_resp / 100,
                      motor - (distL - distR) * dist_diff_resp / 100)
                else (0, 0)
```

Emfrpによる倒立振子ロボットの制御プログラムの例

[1] Sawada, K. & T. Watanabe, "Emfrp: A Functional Reactive Programming Language for Small-Scale Embedded Systems", CROW '16, ACM, 2016.

進行中の関連プロジェクト（抜粋）

EvEmfrp　Emfrpではクロック（あるは周期的パルス）を入力時変値とすることで時間依存動作を記述できますが，その場合の実行効率に課題があります．EvEmfrp[2]は，Emfrpに周期的および非周期的タスクに関する拡張を行った言語です．この言語では時変値の更新タイミングを表現する型とタイミング間の演算を導入することで，周期的・非周期的タスクのより簡潔な記述，および効率的な実行が可能になります．

[2] Sogo et al., “Periodic and Aperiodic Task Description Mechanisms in an FRP Language for Small-Scale Embedded Systems”, REBLS '23., ACM, 2023.

ULPコプロセッサの活用　最近のマイクロコントローラでは，メインとなるCPUの他に機能が限定された省電力(ULP)コプロセッサを持つものがあり，例えばメインCPUを止めてULPコプロセッサで入力を監視することでバッテリーの消費を抑えるといった使い方ができます．我々は状態遷移モデルを導入したFRP言語によってメインCPU上とULPコプロセッサ上の動作が混在するプログラムをシームレスに行う手法[3]を提案し，面倒なULPコプロセッサのプログラミングを容易にすることに成功しています．

[3] Suzuki et al., “Using Low Power Coprocessors in an FRP Language for Embedded Systemss”, APRIS '23., IPSJ, 2023.