特設

第4章 TCP/UDPでおなじみの定番 Socket APIでCANが使える

ラズパイ&Linuxで動かせる CAN通信スタックSocketCAN

濱邊 真也

表1 UDP Socket APIとSocketCAN APIの比較

	TCP/UDP Socket API	Socket CAN API

変数	`int socket_fd; //ディスクリプタ`
	`struct sockaddr_in addr; //Ethernet構造体`
		`int socket_fd; //ディスクリプタ`
		`struct sockaddr_can addr; //CANデータ構造体`

ソケット生成	`if((socket_fd = socket(PF_INET, SOCK_DGRAM, 0)) < 0) {/**/}`	`if((socket_fd = socket(PF_CAN, SOCK_RAW, CAN_RAW)) < 0) {/**/}`

NIC指定	`strcpy(ifr.ifr_name,"eth0");`
	`if(ioctl(socket_fd,SIOCGIFINDEX, &ifr) < 0)`
	`{`
	`  //エラー処理`
	`}`
		`strcpy(ifr.ifr_name, "can0" );`
		`if(ioctl(socket_fd, SIOCGIFINDEX, &ifr)<0)`
		`{`
		`  //エラー処理`
		`}`

設定	`memset(&addr, 0, sizeof(addr));`
	`addr.sin_family = AF_INET;`
	`addr.sin_addr.s_addr = htonl(INADDR_ANY);`
	`addr.sin_port = htons(servPort);`
		`memset(&addr, 0, sizeof(addr));`
		`addr.can_family = AF_CAN;`
		`addr.can_ifindex = ifr.ifr_ifindex;`

バインド	`if (bind(socket_fd,&addr, sizeof(addr)) < 0) {`
	`  //エラー処理`
	`}`
		`if (bind(socket_fd, (struct sockaddr *)&addr, sizeof(addr)) < 0) {`
		`  //エラー処理`
		`}`

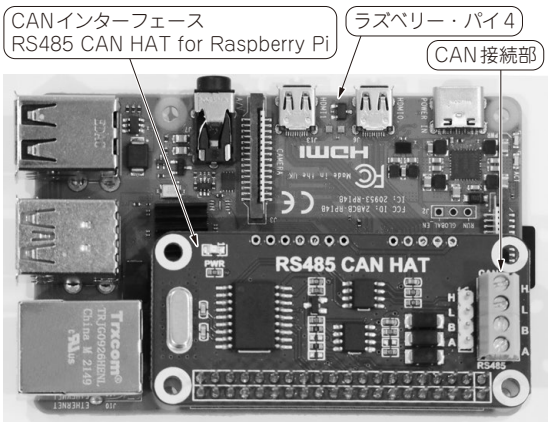

写真1 ラズベリー・パイにCANインターフェースRS485 CAN HATを接続した様子

本稿では，入手性の高いラズベリー・パイ 4と幾つかのCANインターフェースを用いて，SocketCAN環境の構築方法とSocketCAN APIの利用法を紹介します(**写真1**)．

近年はハードウェアにラズベリー・パイなどの安価なシングルボード・コンピュータが利用でき利用になりました．Linux向けの豊富なオープンソース・ソフトウェアも利用できることで，高度な制御を要するFA機器やロボットなどを開発するハードルが低くなっています．

このようなシステム内の通信には，信頼性やリアルタイム性が必要なため，CAN通信を利用したいところです．しかし，CAN通信をLinux機器から利用するには，CANインタフェースの仕様を理解し，専用APIを用いてアプリケーションを開発する必要がありました．

そこで本章では，CANインタフェースに依存せず，UNIX Socket感覚でCAN通信が利用できるSocketCANを紹介します．