목차

Configuration API

시스템 구성 정보를 읽고 쓸 수 있는 API입니다.

- BS2_ResetConfig: 장치의 설정 정보를 초기화합니다.
- BS2_ResetConfigExceptNetInfo: 장치의 설정 정보를 초기화합니다. (네트워크 설정 제외)
- BS2_GetConfig: Configuration blob을 가져옵니다.
- BS2_SetConfig: Configuration blob을 저장합니다.
- BS2_GetFactoryConfig: 소프트웨어 버전 및 하드웨어 정보를 가져옵니다.
- BS2_GetSystemConfig: 시스템 설정 정보를 가져옵니다.
- BS2_SetSystemConfig: 시스템 설정 정보를 저장합니다.
- BS2_GetAuthConfig: 인증 설정 정보를 가져옵니다.
- BS2_SetAuthConfig: 인증 설정 정보를 저장합니다.
- BS2_GetStatusConfig: led, buzzer 설정 정보를 가져옵니다.
- BS2_SetStatusConfig: led, buzzer 설정 정보를 저장합니다.
- BS2_GetDisplayConfig: 효과음과 UI 설정 정보를 가져옵니다.
- BS2_SetDisplayConfig: 효과음과 UI 설정 정보를 저장합니다.
- BS2_GetIPConfig: IP 설정 정보를 가져옵니다.
- BS2_GetIPConfigViaUDP: IP 설정 정보를 UDP broadcasting를 통해 가져옵니다.
- BS2_SetIPConfig: IP 설정 정보를 저장합니다.
- BS2_SetIPConfigViaUDP: IP 설정 정보를 UDP broadcasting를 통해 저장합니다.
- BS2_GetIPConfigExt: DNS 및 Server URL 설정 정보를 가져옵니다.
- BS2_SetIPConfigExt: DNS 및 Server URL 설정 정보를 저장합니다.
- BS2_GetTNAConfig: TNA 설정 정보를 가져옵니다.
- BS2_SetTNAConfig: TNA 설정 정보를 저장합니다.
- BS2_GetCardConfig: 스마트 카드 설정 정보를 가져옵니다.
- BS2_SetCardConfig: 스마트 카드 설정 정보를 저장합니다.
- BS2_GetFingerprintConfig: 지문 매치와 관련된 설정 정보를 가져옵니다.
- BS2_SetFingerprintConfig: 지문 매치와 관련된 설정 정보를 저장합니다.
- BS2_GetRS485Config: RS485 네트워크 설정 정보를 가져옵니다.
- BS2_SetRS485Config: RS485 네트워크 설정 정보를 저장합니다.
- BS2_GetWiegandConfig: Wiegand 입출력 설정 정보를 가져옵니다.
- BS2_SetWiegandConfig: Wiegand 입출력 설정 정보를 저장합니다.
- BS2_GetWiegandDeviceConfig: Wiegand 신호 설정 정보를 가져옵니다.
- BS2_SetWiegandDeviceConfig: Wiegand 신호 설정 정보를 저장합니다.
- BS2_GetInputConfig: Suprevised 입력 포트 설정 정보를 가져옵니다.
- BS2_SetInputConfig: Supervised 입력 포트 설정 정보를 저장합니다.
- BS2_GetWlanConfig: 무선랜 설정 정보를 가져옵니다.
- BS2_SetWlanConfig: 무선랜 설정 정보를 저장합니다.
- BS2_GetTriggerActionConfig: Trigger action 설정 정보를 가져옵니다.
- BS2_SetTriggerActionConfig: Trigger action 설정 정보를 저장합니다.
- BS2_GetEventConfig: Image log filter 설정 정보를 가져옵니다.
- BS2_SetEventConfig: Image log filter 설정 정보를 저장합니다.
- BS2_GetWiegandMultiConfig: WiegandMulti 신호 설정 정보를 가져옵니다.
- BS2_SetWiegandMultiConfig: WiegandMulti 입출력 설정 정보를 저장합니다.
- BS2_GetCard1xConfig: V1.x 제품에서 발급한 Template On Card를 읽을 수 있는 포맷 정보를 저장할 수 있는 설정을 가져옵니다.
- BS2_SetCard1xConfig: V1.x 제품에서 발급한 Template On Card를 읽을 수 있는 포맷 설정 정보를 저장합니다.
- BS2_GetSystemExtConfig: Master와 Slave 장치간의 암호 설정 정보를 가져옵니다.

- BS2_SetSystemExtConfig: Master와 Slave 장치간의 암호 설정 정보를 저장합니다.
- BS2_GetVoipConfig: Voip 설정 정보를 가져옵니다.
- BS2_SetVoipConfig: Voip 설정 정보를 저장합니다.
- BS2_GetFaceConfig: Face 설정 정보를 가져옵니다.
- BS2_SetFaceConfig: Face 설정 정보를 저장합니다.
- BS2_GetRS485ConfigEx: CoreStation일 경우에 RS485 네트워크 설정 정보를 가져옵니다.
- BS2_SetRS485ConfigEx: CoreStation일 경우에 RS485 네트워크 설정 정보를 저장합니다.
- BS2_GetCardConfigEx: iClass SEOS 카드 설정 정보를 가져옵니다.
- BS2_SetCardConfigEx: iClass SEOS 카드 설정 정보를 저장합니다.
- BS2_GetDstConfig: DST 설정 정보를 가져옵니다.
- BS2_SetDstConfig: DST 설정 정보를 저장합니다.
- BS2_GetSupportedConfigMask: 장치에서 지원하는 구성 설정을 가져옵니다.
- BS2_GetIPConfigViaUDPEx: [+ 2.6.3] IP 설정 정보를 UDP broadcasting를 통해 가져옵니다. (host ip 지정)
- BS2_SetIPConfigViaUDPEx: [+ 2.6.3] IP 설정 정보를 UDP broadcasting를 통해 저장합니다. (host ip 지정)
- BS2_GetIPV6Config: [+ 2.6.3] IP V6 설정 정보를 가져옵니다.
- BS2_SetIPV6Config: [+ 2.6.3] IP V6 설정 정보를 저장합니다.
- BS2_GetIPV6ConfigViaUDP: [+ 2.6.3] IP V6 설정 정보를 UDP broadcasting를 통해 가져옵니다.
- BS2_SetIPV6ConfigViaUDP: [+ 2.6.3] IP V6 설정 정보를 UDP broadcasting를 통해 저장합니다.
- BS2_GetIPV6ConfigViaUDPEx: [+ 2.6.3] IP V6 설정 정보를 UDP broadcasting를 통해 가져옵니다. (host ip 지정)
- BS2_SetIPV6ConfigViaUDPEx: [+ 2.6.3] IP V6 설정 정보를 UDP broadcasting를 통해 저장합니다. (host ip 지정)
- BS2_GetDesFireCardConfigEx: [+ 2.6.4] DesFire 카드 고급설정 정보를 가져옵니다.
- BS2_SetDesFireCardConfigEx: [+ 2.6.4] DesFire 카드 고급설정 정보를 저장합니다.
- BS2_GetAuthConfigExt: [+ 2.7.1] Visual Face 기반 장치 인증 설정 정보를 가져옵니다.
- BS2_SetAuthConfigExt: [+ 2.7.1] Visual Face 기반 장치 인증 설정 정보를 저장합니다.
- BS2_GetFaceConfigExt: [+ 2.7.1] 열화상 감지와 마스크 착용 체크를 위한 설정 정보를 가져옵니다.
- BS2_SetFaceConfigExt: [+ 2.7.1] 열화상 감지와 마스크 착용 체크를 위한 설정 정보를 저장합니다.
- BS2_GetThermalCameraConfig: [+ 2.7.1] 열화상 카메라 설정을 가져옵니다.
- BS2_SetThermalCameraConfig: [+ 2.7.1] 열화상 카메라 설정을 저장합니다.
- BS2_GetBarcodeConfig: [+ 2.8] XS2 Barcode 관련 설정을 가져옵니다.
- BS2_SetBarcodeConfig: [+ 2.8] XS2 Barcode 관련 설정을 저장합니다.
- BS2_GetInputConfigEx: [+ 2.8.1] IM-120 Input 관련 확장형 설정을 가져옵니다.
- BS2_SetInputConfigEx: [+ 2.8.1] IM-120 Input 관련 확장형 설정을 저장합니다.
- BS2_GetRelayActionConfig: [+ 2.8.1] IM-120 RelayAction 관련 설정을 가져옵니다.
- BS2_SetRelayActionConfig: [+ 2.8.1] IM-120 RelayAction 관련 설정을 저장합니다.
- BS2_GetVoipConfigExt: [+ 2.8.3] 확장형 VoIP 설정을 가져옵니다.
- BS2_SetVoipConfigExt: [+ 2.8.3] 확장형 VoIP 설정을 저장합니다.
- BS2_GetRtspConfig: [+ 2.8.3] RTSP 설정을 가져옵니다.
- BS2_SetRtspConfig: [+ 2.8.3] RTSP 설정을 저장합니다.
- BS2_GetLicenseConfig: [+ 2.9.1] 장치 라이선스 활성화 정보를 가져옵니다.
- BS2_GetOsdpStandardConfig: [+ 2.9.1] 장치의 OSDP 설정 정보를 가져옵니다.
- BS2_GetOsdpStandardActionConfig: [+ 2.9.1] OSDP 장치의 action별 LED/buzzer 설정을 가져옵니다.
- BS2_SetOsdpStandardActionConfig: [+ 2.9.1] OSDP 장치의 action별 LED/buzzer 설정을 저장합니다.
- BS2_GetCustomCardConfig: [+ 2.9.4] Custom smart card 설정을 가져옵니다.
- BS2_SetCustomCardConfig: [+ 2.9.4] Custom smart card 설정을 저장합니다.

- BS2_GetMifareCardConfigEx: [+ 2.9.9] 장치의 Mifare 카드 중 AES128 암호화 된 카드에 사용되는 설정 정보를 가져옵니다.
- BS2_SetMifareCardConfigEx: [+ 2.9.9] 장치의 Mifare 카드 중 AES128 암호화 된 카드에 사용되는 설정 정보를 저장합니다.

구조체

BS2FactoryConfig

```
typedef struct {
    uint8_t major;
    uint8_t minor;
    uint8_t ext;
    uint8_t reserved[1];
} Version;

typedef struct {
    uint32_t deviceID;
    uint8_t macAddr[BS2_MAC_ADDR_LEN];
    uint8_t reserved[2];
    char modelName[BS2_MODEL_NAME_LEN];
    Version boardVer;
    Version kernelVer;
    Version bscoreVer;
    Version firmwareVer;
    char kernelRev[BS2_KERNEL_REV_LEN];
    char bscoreRev[BS2_BSCORE_REV_LEN];
    char firmwareRev[BS2_FIRMWARE_REV_LEN];
    uint8_t reserved2[32];
} BS2FactoryConfig;
```

1. *deviceID*
장치 식별자입니다.

2. *macAddr*
네트워크 어댑터의 물리적 주소입니다.

3. *reserved*
예약된 공간입니다.

4. *modelName*
장치 모델 이름입니다.

5. *boardVer*
장치 하드웨어 버전입니다.

6. *kernelVer*
장치 커널 버전입니다.

7. *bscoreVer*
장치 BioStar core 버전입니다.

8. *firmwareVer*
장치 펌웨어 버전입니다.

9. *kernelRev*
장치 커널 개정판 정보입니다.

10. *bscoreRev*
장치 BioStar core 개정판 정보입니다.

11. *firmwareRev*
장치 펌웨어 개정판 정보입니다.

12. *reserved2*
예약된 공간입니다.

BS2SystemConfig

```
typedef struct {
    uint8_t notUsed[16 * 16 * 3];
    int32_t timezone;
    uint8_t syncTime;
    uint8_t serverSync;
    uint8_t deviceLocked;
    uint8_t useInterphone;
    uint8_t useUSBConnection;
    uint8_t keyEncrypted;
    uint8_t useJobCode;
    uint8_t useAlphanumericID;
    uint32_t cameraFrequency;
    bool secureTamper;
    bool reserved0;     // (write protected)
    uint8_t reserved[2];
    uint32_t useCardOperationMask;
    uint8_t reserved2[16];
} BS2SystemConfig;
```

1. *notUsed*
사용하지 않는 변수입니다.

2. *timezone*
표준 시간대를 나타내며 단위는 초(sec)를 사용합니다.

3. *syncTime*
BioStar 애플리케이션과 시간 동기화를 결정하는 flag입니다.

4. *serverSync*
예약된 변수입니다.

5. *deviceLocked*
장치의 현재 잠김상태를 나타내는 필드입니다. (읽기전용 필드)

6. *useInterphone*

인터폰을 사용할지 결정하는 flag입니다.

7. *useUSBConnection*
사용되지 않는 member 변수입니다. (USB 연결은 장치가 자동으로 감지합니다.)

8. *keyEncrypted*
OSDP secure key를 사용할지 flag입니다.

9. *useJobCode*
Job code를 사용할지 결정하는 flag입니다.

10. *useAlphanumericID*
AlphanumericID를 사용할지 결정하는 flag입니다.

11. *cameraFrequency*
camera 주파수입니다.

값	설명
1	50Hz
2	60Hz

12. *secureTamper*
보안 탬퍼를 사용할지 결정하는 flag입니다.
탬퍼 on 발생 시 다음의 데이터가 장치에서 삭제됩니다. (사용자, 로그, 데이터 암호화 키, SSL 인증서)

13. *reserved0*
예약된 공간입니다.

14. *reserved*
예약된 공간입니다.

15. *useCardOperationMask*
[+ 2.6.4] 모든 카드를 읽지 않고, 사용자 선택에 의해 카드 타입별로 읽을 수 있도록 옵션을 제공합니다.
MASK값은 조합을 통하여 중복 설정이 가능하며, 사용자는 이 옵션을 통하여 카드 타입의 읽기를 추가 또는 제거할 수 있습니다.
단, 장치가 지원할 수 있는 카드 타입에 한하며, 장치가 지원하지 않는 카드 읽기를 추가하는 경우 해당 설정은 무시됩니다.
또한, 지원하고자하는 카드 타입의 마스크는 `CARD_OPERATION_USE`와 조합해야 합니다.
예를들어 EM카드만 읽을 수 있도록 설정하고자 한다면 useCardOperationMask에는 `0x80000001`로 설정되어져야 합니다.

값	설명
0xFFFFFFFF	CARD_OPERATION_MASK_DEFAULT
0x80000000	CARD_OPERATION_USE
0x00000000	CARD_OPERATION_MASK_NONE
0x00000800	CARD_OPERATION_MASK_CUSTOM_DESFIRE_EV1
0x00000400	CARD_OPERATION_MASK_CUSTOM_CLASSIC_PLUS
0x00000200	CARD_OPERATION_MASK_BLE
0x00000100	CARD_OPERATION_MASK_NFC
0x00000080	CARD_OPERATION_MASK_SEOS
0x00000040	CARD_OPERATION_MASK_SR_SE
0x00000020	CARD_OPERATION_MASK_DESFIRE_EV1

값	설명
0x00000010	CARD_OPERATION_MASK_CLASSIC_PLUS
0x00000008	CARD_OPERATION_MASK_ICLASS
0x00000004	CARD_OPERATION_MASK_MIFARE_FELICA
0x00000002	CARD_OPERATION_MASK_HIDPROX
0x00000001	CARD_OPERATION_MASK_EM

16. *reserved2*
예약된 공간입니다.

BS2AuthConfig

```
typedef struct {
    uint32_t authSchedule[BS2_NUM_OF_AUTH_MODE];
    uint8_t useGlobalAPB;
    uint8_t globalAPBFailAction;
    uint8_t useGroupMatching;
    uint8_t reserved
    uint8_t reserved[28];
    uint8_t usePrivateAuth;
    uint8_t faceDetectionLevel;
    uint8_t useServerMatching;
    uint8_t useFullAccess;
    uint8_t matchTimeout;
    uint8_t authTimeout;
    uint8_t numOperators;
    uint8_t reserved2[1];
    struct {
        char userID[BS2_USER_ID_SIZE];
        uint8_t level;
        uint8_t reserved[3];
    } operators[BS2_MAX_OPERATORS];
} BS2AuthConfig;
```

1. *authSchedule*
각 인증 모드가 활성화될 때 실행할 일정 식별자입니다.
배열의 순서대로 아래의 의미를 가지며,
배열 내의 값이 0보다 큰경우 해당 인증모드는 활성화 되어 있습니다.
아래 설명 중 생체정보는 장치에 따라 지문 또는 얼굴을 의미합니다.

순서	의미	설명
0	BS2_AUTH_MODE_BIOMETRIC_ONLY	생체정보
1	BS2_AUTH_MODE_BIOMETRIC_PIN	생체정보 + PIN번호
2	BS2_AUTH_MODE_CARD_ONLY	카드번호
3	BS2_AUTH_MODE_CARD_BIOMETRIC	카드번호 + 생체정보

순서	의미	설명
4	BS2_AUTH_MODE_CARD_PIN	카드번호 + PIN번호
5	BS2_AUTH_MODE_CARD_BIOMETRIC_OR_PIN	카드번호 + 생체정보 or PIN번호
6	BS2_AUTH_MODE_CARD_BIOMETRIC_PIN	카드번호 + 생체정보 + PIN번호
7	BS2_AUTH_MODE_ID_BIOMETRIC	ID + 생체정보
8	BS2_AUTH_MODE_ID_PIN	ID + PIN 번호
9	BS2_AUTH_MODE_ID_BIOMETRIC_OR_PIN	ID + 생체정보 or PIN번호
10	BS2_AUTH_MODE_ID_BIOMETRIC_PIN	ID + 생체정보 + PIN번호

2. *useGlobalAPB*
광역 이중 인증 방지 구역을 활성화할지 결정하는 flag입니다.

3. *globalAPBFailAction*
광역 이중 인증 위반인지 BioStar 애플리케이션에 질의를 할 수 없거나 응답이 없을때 기본적으로 수행해야 할 동작입니다.

값	설명
0	APB 검사 안함
1	Soft APB로 동작
2	Hard APB로 동작

4. *useGroupMatching*
얼굴 그룹 매칭을 사용할지 결정하는 flag입니다.

5. *reserved*
예약된 공간입니다.

6. *usePrivateAuth*
개별 인증 모드를 사용할지 결정하는 flag입니다.

7. *faceDetectionLevel*
A2에서 사용자를 인증할 때 얼굴 검출 레벨값이며, 지정한 레벨보다 낮은 레벨로 얼굴이 검출되면 인증 실패로 처리됩니다.
설정하면 Normal/Strict에 따른 카메라 뷰가 표시되며, 인증 성공 시의 이미지 로그가 얼굴로 인식되지 않으면 접근이 거부됩니다. 기본값은 0입니다.

값	설명
0	얼굴 검출을 하지 않음

값	설명
1	Normal mode
2	Strict mode

A2에서만 설정 가능하며, FaceStation2나 FaceLite에서는 사용되지 않습니다.

8. *useServerMatching*
지문 인증이나 얼굴 인식을 Matching server에서 수행할지 결정하는 flag입니다.

9. *useFullAccess*
사용하지 않는 변수입니다.

10. *matchTimeout*
지문 인증이나 얼굴 인식에서 최대 응답 시간이며 단위는 초(sec)를 사용합니다.

11. *authTimeout*
사용자 인증의 최대 응답 시간이며 단위는 초(sec)를 사용합니다.

12. *numOperators*
사용자 권한을 정의하는 operator의 개수입니다.

13. *reserved2*
예약된 공간입니다.

14. *userID*
사용자 식별자입니다.

15. *level*
사용자가 인증되었을 때 식별자에 해당하는 권한을 지정합니다.

값	설명
0	권한 없음
1	관리자 권한
2	시스템 구성을 변경할 수 있는 권한
3	사용자 정보를 변경할 수 있는 권한

주의
Operator 추가 시, 추가하고자 하는 operator의 수를 ***numOperators*** 필드의 값으로 지정해야 합니다.

16. *reserved*
예약된 공간입니다.

BS2StatusConfig

```
typedef struct {
    struct {
        uint8_t enabled;
        uint8_t reserved[1];
        uint16_t count;
        BS2LedSignal signal[BS2_LED_SIGNAL_NUM];
    } led[BS2_DEVICE_STATUS_NUM];
    uint8_t reserved1[32];
    struct {
        uint8_t enabled;
        uint8_t reserved[1];
        uint16_t count;
        BS2BuzzerSignal signal[BS2_BUZZER_SIGNAL_NUM];
    } buzzer[BS2_DEVICE_STATUS_NUM];
    uint8_t configSyncRequired;
    uint8_t reserved2[31];
} BS2StatusConfig;
```

1. *enabled*
led를 사용할지 결정하는 flag입니다.

2. *reserved*
예약된 공간입니다.

3. *count*
led signal의 수행 횟수이며, 0으로 설정시 무한 반복됩니다.

4. *signal*
led signal pattern을 나열한 리스트이며, 최대 3개까지 설정할 수 있습니다.

5. *reserved1*
예약된 공간입니다.

6. *enabled*
buzzer를 사용할지 결정하는 flag입니다.

7. *reserved*
예약된 공간입니다.

8. *count*
buzzer signal의 수행 횟수이며, 0으로 설정시 무한 반복됩니다.

9. *signal*
buzzer signal pattern을 나열한 리스트이며, 최대 3개까지 설정할 수 있습니다. 10. *configSyncRequired*
장치를 조작하여 configuration이 변경되었다면, 이 값이 true로 설정됩니다.

11. *reserved2*
예약된 공간입니다.

BS2DisplayConfig

```
typedef struct {
    uint32_t language;
    uint8_t background;
    uint8_t volume;
    uint8_t bgTheme;
    uint8_t dateFormat;
    uint16_t menuTimeout;
    uint16_t msgTimeout;
    uint16_t backlightTimeout;
    uint8_t displayDateTime;
    uint8_t useVoice;
    uint8_t timeFormat;
    uint8_t homeFormation;
    BS2_BOOL useUserPhrase;
    BS2_BOOL queryUserPhrase;
    uint8_t shortcutHome[BS2_MAX_SHORTCUT_HOME];
    uint8_t tnaIcon[16];
    uint8_t useScreenSaver;
    uint8_t showOsdpResult;
    uint8_t reserved1[30];
    BS2_SHOW_OSDP_RESULT showOsdpResult;   ///< 1 byte

    BS2_AUTHMSG_USERINFO authMsgUserName;          ///< 1 byte
    BS2_AUTHMSG_USERINFO authMsgUserId;            ///< 1 byte

    BS2_SCRAMBLE_KEYBOARD_MODE scrambleKeyboardMode ;      ///< 1 byte

    uint8_t reserved3[27];     ///< 27 bytes (padding)
} BS2DisplayConfig;
```

1. *language*
언어 설정 정보입니다.

값	설명
0	한국어
1	영어
2	사용자 정의

2. *background*
배경 이미지 설정 정보입니다.

값	설명
0	LOGO
1	NOTICE
2	SLIDE
3	PDF

3. *volume*
효과음의 소리 크기이며 0-100 단계로 설정할 수 있습니다. 0으로 설정하면 효과음이 들리지 않습니다.

4. *bgTheme*
테마 설정 정보입니다.

값	설명
0	로고 이미지
1	공지 메시지
2	Slide show
3	PDF

5. *dateFormat*
날짜 형식 설정 정보입니다.

값	설명
0	YYYY/MM/DD
1	MM/DD/YYYY
2	DD/MM/YYYY

6. *menuTimeout*
사용자의 장치 조작이 없을 때 잠금 화면으로 전환되는 시간이며 단위는 초(sec)를 사용합니다. 0-255 sec로 설정할 수 있습으며. 0으로 설정되면 잠금 화면으로 전환되지 않습니다.

값	설명
0	메뉴화면 유지 지속
10	메뉴화면 유지 10초
20	메뉴화면 유지 20초 (기본값)
30	메뉴화면 유지 30초
40	메뉴화면 유지 40초
50	메뉴화면 유지 50초
60	메뉴화면 유지 60초

7. *msgTimeout*
메시지 화면이 자동으로 사라지는 시간이며 단위는 밀리초(ms)를 사용합니다. 500-5000 ms로 설정할 수 있습니다.

값	설명
500	메시지 화면 유지 500밀리초
1000	메시지 화면 유지 1초
2000	메시지 화면 유지 2초 (기본값)
3000	메시지 화면 유지 3초
4000	메시지 화면 유지 4초
5000	메시지 화면 유지 5초

8. *backlightTimeout*
백라이트 유지 시간이며 단위는 초(sec)를 사용합니다.

값	설명
0	백라이트 유지 0초
10	백라이트 유지 10초
20	백라이트 유지 20초 (기본값)
30	백라이트 유지 30초
40	백라이트 유지 40초
50	백라이트 유지 50초

값	설명
60	백라이트 유지 60초

9. *displayDateTime*
화면에 시간을 표시할지 결정하는 flag입니다.

10. *useVoice*
voice instruction을 사용할지 결정하는 flag입니다.

11. *timeFormat*
시간 형식 설정 정보입니다.

값	설명
0	12시간 단위 표시
1	24시간 단위 표시

그러나, Linux 제품군인 BioStation 2, BioStation L2, BioLite Net2, FaceLite는 반대의 설정값을 가지고 있습니다. (0 = 24 hour / 1 = 12 hour)

12. *homeFormation*
Home 화면 설정 정보로 추후 적용 예정입니다.

값	설명
1	인터폰
2	Shortcut 1
3	Shortcut 2
4	Shortcut 3
5	Shortcut 4

13. *useUserPhrase*
인증시 개인메시지 표시 여부입니다.

14. *queryUserPhrase*
true로 설정 시, 개인인증 메시지를 서버에 묻습니다.

15. *shortcutHome*
homeFormation과 관련된 변수로 추후 적용 예정입니다.

16. *tnaIcon*
장치내에서 근태 코드값에 해당하는 아이콘을 보여줄 때 사용합니다.

17. *useScreenSaver*
true로 설정 시, 화면보호기가 활성화됩니다.

18. *showOsdpResult*
[+ 2.9.6] Intelligent slave나 타사 컨트롤러에 슈프리마 장치를 주변장치로 연결하여 사용하는 환경에서 인증 결과를 장치 화면에 표시하도록 하는 옵션입니다.

값	설명
0	OSDP 인증결과 표시 (기본값)

값	설명
1	OSDP 인증결과 표시안함

19. *reserved1*
예약된 공간입니다.

20. *authMsgUserName*
[+2.9.8] 장치 화면에 표시되는 인증 결과 메시지에 포함된 사용자 이름의 표시 방법을 설정합니다.

값	설명
0	전체 표시
1	첫 문자만 표시
2	표시 안함

21. *authMsgUserID*
[+2.9.8] 장치 화면에 표시되는 인증 결과 메시지에 포함된 사용자 ID의 표시 방법을 설정합니다.

값	설명
0	전체 표시
1	첫 문자만 표시
2	표시 안함

22. *scrambleKeyboardMode*
[+2.9.8] 스크램블 키패드의 사용 여부를 설정합니다.

값	설명
0	사용
1	사용 안함

BS2IpConfig

```
typedef struct {
    uint8_t connectionMode;
    uint8_t useDHCP;
    uint8_t useDNS;
    uint8_t reserved[1];
    char ipAddress[BS2_IPV4_ADDR_SIZE];
    char gateway[BS2_IPV4_ADDR_SIZE];
    char subnetMask[BS2_IPV4_ADDR_SIZE];
    char serverAddr[BS2_IPV4_ADDR_SIZE];
    uint16_t port;
    uint16_t serverPort;
    uint16_t mtuSize;
    uint8_t baseband;
    uint8_t reserved2[1];
    uint16_t sslServerPort
    uint8_t reserved3[30];
} BS2IpConfig;
```

1. *connectionMode*

BioStar 애플리케이션과 장치간의 접속모드를 뜻하며, 접속 주체에 따라 direct mode(0x0)와 server mode(0x1)로 구분됩니다. direct mode는 BioStar 애플리케이션이 장치로 접속하며 server mode는 장치가 BioStar 애플리케이션으로 접속합니다. 장치에 설정된 기본 접속 모드는 direct mode입니다.

2. *useDHCP*
DHCP 사용 유무를 나타내는 flag입니다.

3. *useDNS*
server addresss를 사용할지 server URL을 사용할 지 결정하는 flag입니다.

4. *reserved*
예약된 공간입니다.

5. *ipAddress*
현재 장치의 IP 주소입니다.

6. *gateway*
현재 장치의 게이트웨이 IP 주소입니다.

7. *subnetMask*
현재 장치의 서브넷 마스크입니다.

8. *serverAddr*
connectionMode가 server mode일 때 사용되며, BioStar 애플리케이션의 IP 주소입니다.

9. *port*
현재 장치의 IP 포트 번호입니다.

10. *serverPort*
connectionMode가 server mode일 때 사용되며, BioStar 애플리케이션의 포트 번호입니다.

11. *mtuSize*
TCP 통신에 사용할 MTU[1] 크기입니다.

12. *baseband*
장치의 baseband이며 10mb/s나 100mb/s의 속도를 가집니다.

13. *reserved2*
예약된 공간입니다.

14. *sslServerPort*
connectionMode가 server ssl mode일 때 사용되며, BioStar 애플리케이션의 포트 번호입니다.

15. *reserved3*
예약된 공간입니다.

BS2IpConfigExt

```
typedef struct {
    char dnsAddr[BS2_IPV4_ADDR_SIZE];
    char serverUrl[BS2_URL_SIZE];
    uint8_t reserved[32];
} BS2IpConfigExt;
```

1. *dnsAddr*
dns 서버 주소입니다.

2. *serverUrl*
BioStar 애플리케이션 서버 URL이며, 최대 256자까지 입력할 수 있습니다.

3. *reserved*
예약된 공간입니다.

BS2TNAConfig

```
typedef struct {
    uint8_t tnaMode;
    uint8_t tnaKey;
    uint8_t tnaRequired;
    uint8_t reserved[1];
    uint32_t tnaSchedule[BS2_MAX_TNA_KEY];
    uint8_t unused[BS2_MAX_TNA_KEY];
} BS2TNAInfo;

typedef struct {
    char tnaLabel[BS2_MAX_TNA_KEY][BS2_MAX_TNA_LABEL_LEN];
    uint8_t unused[BS2_MAX_TNA_KEY];
} BS2TNAExtInfo;

typedef struct {
    BS2TNAInfo tnaInfo;
    BS2TNAExtInfo tnaExtInfo;
    uint8_t reserved2[32];
} BS2TNAConfig;
```

1. *tnaMode*
근태 관리 모드입니다.

값	설명
0	사용안함
1	사용자에 의한 근태 코드 적용
2	근태 관리 일정에 맞춰 근태 코드 적용
3	사용자가 마지막으로 선택한 근태 코드 적용
4	고정된 근태 코드 사용

2. *tnaKey*
근태 코드값을 의미하는 단축키이며, 장치의 키패드에 맵핑됩니다.

Device Type	T&A Code	Mapped Key	Value
BioStation 2	BS2_TNA_UNSPECIFIED	(N/A)	0
	BS2_TNA_KEY_1	F1	1
	BS2_TNA_KEY_2	F2	2
	BS2_TNA_KEY_3	F3	3
	BS2_TNA_KEY_4	F4	4
	BS2_TNA_KEY_5	1	5
	BS2_TNA_KEY_6	2	6
	BS2_TNA_KEY_7	3	7
	BS2_TNA_KEY_8	4	8
	BS2_TNA_KEY_9	5	9
	BS2_TNA_KEY_10	6	10
	BS2_TNA_KEY_11	7	11
	BS2_TNA_KEY_12	8	12
	BS2_TNA_KEY_13	9	13
	BS2_TNA_KEY_14	Call	14
	BS2_TNA_KEY_15	0	15
	BS2_TNA_KEY_16	Esc	16

3. *tnaRequired*
근태 모드가 1로 설정되었을 경우에 근태 코드값이 요구되는지 결정하는 flag입니다.

4. *reserved*
예약된 공간입니다.

5. *tnaSchedule*
각 근태 코드가 수행할 일정을 지정합니다.

6. *unused*
사용하지 않는 변수입니다.

7. *tnaLabel*
근태 코드값이 무엇을 의미하는지 나타내는 별칭입니다.

8. *unused*
사용하지 않는 변수입니다.

BS2CardConfig

```
typedef struct {
    uint8_t primaryKey[6];
    uint8_t reserved1[2];
    uint8_t secondaryKey[6];
    uint8_t reserved2[2];
    uint16_t startBlockIndex;
    uint8_t reserved[6];
} BS2MifareCard;
```

```
typedef struct {
    uint8_t primaryKey[8];
    uint8_t secondaryKey[8];
    uint16_t startBlockIndex;
    uint8_t reserved[6];
} BS2IClassCard;

typedef struct {
    uint8_t primaryKey[16];
    uint8_t secondaryKey[16];
    uint8_t appID[3];
    uint8_t fileID;
    uint8_t encryptionType;
    uint8_t operationMode;
    uint8_t reserved[2];
} BS2DesFireCard;

typedef struct {
    uint8_t byteOrder;
    uint8_t useWiegandFormat;
    uint8_t dataType;
    uint8_t useSecondaryKey;
    BS2MifareCard mifare;
    BS2IClassCard iclass;
    BS2DesFireCard desfire;
    uint8_t formatID;
    uint8_t cipher;
    uint8_t smartCardByteOrder;
    uint8_t reserved[1];
BS2_MIFARE_ENCRYPTION mifareEncType;
uint8_t reserved[20];
} BS2CardConfig;
```

1. *primaryKey*
Mifare card 정보에 접근하기 위한 첫번째 암호화 키입니다.

2. *reserved1*
예약된 공간입니다.

3. *secondaryKey*
Mifare card 정보에 접근하기 위한 두번째 암호화 키입니다.

4. *reserved2*
예약된 공간입니다.

5. *startBlockIndex*
Mifare data storage에서의 start block index입니다.

6. *reserved*
예약된 공간입니다.

7. *primaryKey*
IClass card 정보에 접근하기 위한 첫번째 암호화 키입니다.

8. *secondaryKey*
IClass card 정보에 접근하기 위한 두번째 암호화 키입니다.

9. *startBlockIndex*
Mifare data storage에서의 start block index입니다.

10. *reserved*
예약된 공간입니다.

11. *primaryKey*
DesFire card 정보에 접근하기 위한 첫번째 암호화 키입니다.

12. *secondaryKey*
DesFire card 정보에 접근하기 위한 두번째 암호화 키입니다.

13. *appID*
사용자 인증을 위해 DESFire 카드내에 내장된 사용할 애플리케이션 식별자입니다.

14. *fileID*
DESFire 카드내에 내장된 사용할 애플리케이션가 데이터를 읽고 쓰기 위한 파일 식별자입니다.

15. *encryptionType*
데이터 암호화 방식을 나타냅니다.

값	설명
0	DES/3DES
1	AES

16. *operationMode*
운영 방식을 지정합니다. (추후적용)

값	설명
0	기존방식 (PICC master key를 사용)
1	신규방식 (App master key를 사용)

17. *reserved*
예약된 공간입니다.

18. *byteOrder*
카드 데이터의 바이트 저장 순서입니다. 0일 경우 MSB[2]이고, 1일 경우 LSB[3]입니다.

19. *useWiegandFormat*
Wiegand 형식 카드 사용 유무를 결정하는 flag입니다.

20. *dataType*
Card의 데이터 유형입니다.

값	설명
0	바이너리
1	아스키

값	설명
2	UTF16
3	BCD

21. *useSecondaryKey*
두번째 암호화 키 사용 유무를 결정하는 flag입니다.

22. *formatID*
BioStar 애플리케이션에서 card configuration을 데이터베이스 관리할 필요가 있을 경우 사용할 수 있는 식별자입니다.

23. *cipher*
Keypad로 card id를 입력받을 수 있도록 활성화 합니다.
기본 값은 0이며, Xpass 2, Xpass D2의 Gangbox Keypad 모델만 유효합니다.

값	설명
0	비활성화
1	활성화

24. *smartCardByteOrder*
[+2.8.2] 이전까지 smart card의 data 영역에 쓰여진 카드정보는 MSB 방식으로 고정이었습니다.
이로인해 LSB 방식을 사용하는 일부 controller로의 연동시, byte 정보가 반전되어 전혀 다른값으로 출력되었습니다.
smartCardByteOrder 값으로, MSB/LSB 출력을 유연하게 변경할 수 있습니다.

값	설명
0	MSB
1	LSB

25. *reserved*
예약된 공간입니다.
26. *mifareEncType*
[+2.9.9] Mifare Card의 암호화 방식을 지정합니다. Mifare Classic 카드는 CRYPTO1만 지원하며, Mifare Plus 카드는 CRYPTO1과 AES128 모두 지원합니다.
CRYPTO1으로 설정하면 BS2MifareCard의 Mifare 설정이 적용되고, AES128로 설정하면 BS2_SetMifareCardConfigEx 및 BS2_GetMifareCardConfigEx를 통해 설정할 수 있는 BS2MifareCardConfigEx 설정이 적용됩니다.

값	설명
0	CRYPTO1
1	AES128

27. *reserved*
예약된 공간입니다.

BS2FingerprintConfig

```
typedef struct {
    uint8_t      securityLevel;
    uint8_t      fastMode;
    uint8_t       sensitivity;
    uint8_t       sensorMode;
    uint16_t      templateFormat;
    uint16_t      scanTimeout;
    uint8_t       successiveScan;
    uint8_t       advancedEnrollment;
    uint8_t       showImage;
    uint8_t       lfdLevel;
    bool          checkDuplicate;

    uint8_t       reserved3[31];
} BS2FingerprintConfig;
```

1. *securityLevel*
시스템 전반적으로 사용할 지문 인증 보안 레벨입니다.

값	설명
0	기본
1	보안 레벨 높음
2	보안 레벨 매우 높음

2. *fastMode*
지문 인증 속도 설정입니다.

값	설명
0	자동
1	기본 인증 속도
2	인증 속도 빠름
3	인증 속도 매우 빠름

3. *sensitivity*
지문 센서의 민감도 설정입니다.

값	설명
0	민감도 가장 낮음
1	민감도 레벨 1
2	민감도 레벨 2
3	민감도 레벨 3
4	민감도 레벨 4
5	민감도 레벨 5
6	민감도 레벨 6
7	민감도 가장 높음

4. *sensorMode*
지문 센서를 어떻게 동작시킬지 결정하는 flag입니다. 0일 경우 센서를 항상 켜지고, 1일 경우 근접센서에 손가락이 접근할 때에만 센서가 켜집니다.

5. *templateFormat*
지문 템플릿 저장 형식입니다.

값	설명
0	suprema 포맷
1	국제 표준 포맷
2	Ansi 포맷

6. *scanTimeout*
지문 스캔 대기 시간으로 기본값은 10초입니다.

7. *successiveScan*
사용하지 않는 변수입니다.

8. *advancedEnrollment*
지문 등록할 때 지문 이미지 품질 정보를 사용할지 결정하는 flag입니다. 만약, 이 옵션이 비활성화되어 있다면 획득한 지문 이미지의 품질이 낮더라도 BS2_ScanFingerprint 함수에서는 BS_SDK_ERROR_EXTRACTION_LOW_QUALITY 또는 BS_SDK_ERROR_CAPTURE_LOW_QUALITY 에러 코드를 반환하지 않습니다.

9. *showImage*
스캔된 지문 이미지를 화면에 보여줄지 결정하는 flag입니다.

10. *lfdLevel*
위조 지문을 판단하는 민감도 설정입니다.

값	설명
0	사용안함
1	민감도 낮음
2	민감도 보통
3	민감도 높음

11. *checkDuplicate*
[+ V2.6.4] true로 설정하면 중복된 지문인지 판단합니다.

12. *reserved3*
예약된 공간입니다.

BS2Rs485Config

```
typedef struct {
    uint8_t supportConfig;
    uint8_t useExceptionCode;
    uint8_t exceptionCode[BS2_RS485_MAX_FAIL_CODE_LEN];
    uint8_t outputFormat;
    uint8_t osdpID;
    uint8_t reserved[4];
} BS2IntelligentPDInfo;

typedef struct {
    uint32_t baudRate;
    uint8_t channelIndex;
    uint8_t useRegistance;
    uint8_t numOfDevices;
    uint8_t reserved[1];
    BS2Rs485SlaveDevice slaveDevices[BS2_RS485_MAX_SLAVES_PER_CHANNEL];
} BS2Rs485Channel;

typedef struct {
    uint8_t mode;
    uint8_t numOfChannels;
    uint8_t reserved[2];
    BS2IntelligentPDInfo intelligentInfo;
    uint8_t reserved1[16];
    BS2Rs485Channel channels[BS2_RS485_MAX_CHANNELS];
} BS2Rs485Config;
```

1. *supportConfig*
`[+V2.8]` 이 값이 0으로 전달되면 장치는 아래의 Intelligent PD(Peripheral Device) 관련 설정들을 모두 무시하게 됩니다.

2. *useExceptionCode*
`[+V2.8]` 예외코드를 전송을 사용할지의 여부를 설정합니다.

3. *exceptionCode*
`[+V2.8]` 인증에 실패하거나, 인증 성공이지만 카드가 등록되어있지 않은 사용자의 경우 예외코드를 내보낼 수 있습니다.
이때 사용할 예외코드를 설정합니다.
만일 예외코드가 0(0x0000000000000000)이라면 예외코드가 발생되지 않습니다.

4. *outputFormat*
`[+V2.8]` 인증 성공 시 내보내어지는 정보를 지정할 수 있습니다.
0 이면 카드ID가, 1 이면 사용자ID가 출력됩니다.

5. *osdpID*
`[+V2.8]` 동일한 ACU에 연결된 장치들을 서로 구분시켜주기위해 사용되어지는 값으로,
0~127 사이의 unique한 값을 설정해 지정하여주면 됩니다.

6. *reserved*
`[+V2.8]` 예약된 공간입니다.

7. *baudRate*
RS485 통신 속도이며 설정할 수 있는 범위는 다음과 같습니다.

값
9600
19200
38400
57600
115200

8. *channelIndex*
(사용자 설정 불가) RS485 network의 통신 채널 번호입니다.

9. *useRegistance*
종단 저항을 설정할지 결정하는 flag입니다. - 동작에 영향 없음

10. *numOfDevices*
슬레이브 장치의 개수입니다.

11. *reserved*
예약된 공간입니다.

12. *slaveDevices*
슬레이브 장치를 나열한 리스트로 최대 32개까지 정의할 수 있습니다.

13. *mode*
RS485 네트워크에서 어떤 모드로 동작할지 결정하는 flag입니다 .

값	설명
1	Master
2	Slave
3	Standalone

14. *numOfChannels*
RS485 채널의 개수입니다.

15. *reserved*
예약된 공간입니다.

16. *intelligentInfo*
`[+V2.8]` Intelligent Slave 기능을 설정하며, mode가 default(Standalone)인 경우에만 동작됩니다.
이 설정을 이용하면 슈프리마 기기가 타사 컨트롤러와 연결된 환경에서 OSDP로 인증 결과를 출력시켜 줍니다.

17. *reserved1*
예약된 공간입니다.

18. *channels*
RS485 채널을 나열한 리스트로 최대 4개까지 설정할 수 있습니다.

BS2WiegandConfig

```
typedef struct {
    uint32_t length;
    uint8_t idFields[BS2_WIEGAND_MAX_FIELDS][BS2_WIEGAND_FIELD_SIZE];
    uint8_t parityFields[BS2_WIEGAND_MAX_PARITIES][BS2_WIEGAND_FIELD_SIZE];
    BS2_WIEGAND_PARITY parityType[BS2_WIEGAND_MAX_PARITIES];
    uint8_t parityPos[BS2_WIEGAND_MAX_PARITIES];
} BS2WiegandFormat;

typedef struct {
    uint8_t mode;
    uint8_t useWiegandBypass;
    uint8_t useFailCode;
    uint8_t failCode;
    uint16_t outPulseWidth;
    uint16_t outPulseInterval;
    uint32_t formatID;
    BS2WiegandFormat format;
    uint16_t wiegandInputMask;
    uint16_t wiegandCardMask;
    uint8_t wiegandCSNIndex;
    uint8_t useWiegandUserID;
    uint8_t reserved[26];
} BS2WiegandConfig;
```

1. *length*
Wiegand 카드 포맷의 길이입니다.

2. *idFields*
최대 4개의 id field를 가질수 있습니다. 각 field마다 id의 시작부터 끝까지 bit를 세팅한 값을 배열의 끝에서부터 입력해야 합니다. 예를들어, Standard 26bit wiegand card data는 “P FFFFFFFF NNNNNNNNNNNNNNNN P” 형태로 구성되어 있습니다. 따라서 Facillity Code는 “ 0 11111111 0000000000000000 0 ” 이므로0x01FE0000 값을 가지며, Card Number는 0x0001FFFE입니다.

```
// for Facillity Code
idFields[][28] = 0x01;
idFields[][29] = 0xFE;
idFields[][30] = 0x00;
idFields[][31] = 0x00;

// for Card Number
idFields[1][28] = 0x00;
idFields[1][29] = 0x01;
idFields[1][30] = 0xFF;
idFields[1][31] = 0xFE;
```

3. *parityFields*
최대 4개의 패리티 필드를 가질수 있으며, 패리티를 체크할 시작과 끝을 id Field와 동일한 방법으로 입력합니다.

4. *parityType*
패리티 유형을 지정합니다.

값	설명
0	parity 체크하지 않음
1	홀수 parity 체크
2	짝수 parity 체크

5. *parityPos*
Wiegand 카드 데이터에서 패리티 값을 저장할 위치를 지정합니다.

6. *mode*
Wiegand 입출력 모드를 설정합니다.

값	설명
0	입력
1	출력
2	입출력

7. *useWiegandBypass*
카드 데이터 출력 여부를 결정하는 플래그입니다.

값	설명
0	인증되면 출력
1	인증없이 출력

8. *useFailCode*
인증되지 않은 카드 데이터를 Fail Code값으로 채워서 보낼지 결정하는 플래그입니다.

9. *failCode*
카드 데이터를 치환할 Fail Code값입니다.

값
0x00
0xFF

10. *outPulseWidth*
출력 진폭값으로 20 ~ 100 us 범위를 가집니다.

11. *outPulseInterval*
출력 주파수값으로 200 ~ 20000 us 범위를 가집니다.

12. *formatID*
애플리케이션에서 Wiegand 카드 포맷을 구별하기 위해 사용하는 값으로 장치에서는 사용하지 않습니다.

13. *format*
WiegandFormat 구조체입니다.

14. *wiegandInputMask*
Master를 제외하고 Slave와 wiegand 장치에서의 wiegand 입력에 대한 mask입니다.

15. *wiegandCardMask*
Master장치에서의 입력일 때의 mask입니다.

16. *wiegandCSNIndex*
Mifare 또는 EM 카드를 사용할 때 Wiegand out을 어떤 형식으로 처리할 것인지 결정하는 인덱스입니다.
설정하기 전에 BS2CardConfig 구조체의 *useWiegandFormat* 값을 확인하십시오.

17. *useWiegandUserID*
Wiegand 출력시 Card ID 또는 사용자 ID를 선택적으로 출력할 수 있도록 결정하는 플래그 입니다.

값	설명
0	사용 안함
1	Card ID
2	사용자 ID

18. *reserved*
예약된 공간입니다.

BS2WiegandDeviceConfig

```
typedef struct {
    uint32_t deviceID;
    uint16_t port;
    uint8_t switchType;
    uint8_t reserved[1];
} BS2WiegandTamperInput;

typedef struct {
    uint32_t deviceID;
    uint16_t port;
    uint8_t reserved[10];
} BS2WiegandLedOutput;

typedef struct {
    uint32_t deviceID;
    uint16_t port;
    uint8_t reserved[34];
} BS2WiegandBuzzerOutput;

typedef struct {
    BS2WiegandTamperInput tamper;
    BS2WiegandLedOutput led[BS2_WIEGAND_STATUS_NUM];
    BS2WiegandBuzzerOutput buzzer;
    uint32_t reserved[32];
} BS2WiegandDeviceConfig;
```

1. *deviceID*
Wiegand card reader의 tamper 신호를 수신받는 장치 식별자입니다.

2. *port*
Wiegand card reader의 tamper 입력 포트입니다.

3. *switchType*
입력 포트의 동작 모드로 대기 상태가 'off'이고 입력 신호가 on일 때 trigger를 발동시킵니다.

값	설명
0	Normally Open
1	Normally Closed

4. *reserved*
예약된 공간입니다.

5. *deviceID*
Wiegand card reader에 led 신호를 출력하는 장치 식별자입니다.

6. *port*
Wiegand card reader의 led 신호 출력 포트입니다.

7. *reserved*
예약된 공간입니다.

8. *deviceID*
Wiegand card reader에 buzzer 신호를 출력하는 장치 식별자입니다.

9. *port*
Wiegand card reader의 buzzer신호 출력 포트입니다.

10. *reserved*
예약된 공간입니다.

10. *led*
Wiegand card reader의 led 신호를 출력하는 장치를 나열한 리스트로 최대 2개까지 설정할 수 있습니다.

값	설명
0	적색 led
1	녹색 led

BS2InputConfig

```
typedef struct {
    uint16_t minValue;
    uint16_t maxValue;
} BS2SVInputRange;

typedef struct {
    uint32_t deviceID;
    uint16_t port;
    uint8_t reserved[10];
} BS2WiegandLedOutput;

typedef struct {
    BS2SVInputRange shortInput;
    BS2SVInputRange openInput;
    BS2SVInputRange onInput;
    BS2SVInputRange offInput;
} BS2SupervisedInputConfig;

typedef struct {
    uint8_t numInputs;
    uint8_t numSupervised;
    union {
    uint16_t value;
    struct {
        uint16_t tamperAuxIndex : 4;
        uint16_t acFailAuxIndex : 4;
        uint16_t aux0Type : 1;
        uint16_t aux1Type : 1;
        uint16_t reserved : 6;
    } field;
    } aux;
    struct {
        uint8_t portIndex;
        uint8_t enabled;
        uint8_t supervised_index;
        uint8_t reserved[5];
        BS2SupervisedInputConfig config;
    } supervised_inputs[BS2_MAX_INPUT_NUM];
} BS2InputConfig;
```

1. *minValue*
최저 전압으로 유효 범위는 0 ~ 3300(3.3v)까지 입니다.

2. *maxValue*
최고 전압으로 유효 범위는 0 ~ 3300(3.3v)까지 입니다.

3. *shortInput*
short input으로 판별할 수 있는 전압 범위입니다.

4. *openInput*
open input으로 판별할 수 있는 전압 범위입니다.

5. *onInput*
on input으로 판별할 수 있는 전압 범위입니다.

6. *offInput*
off input으로 판별할 수 있는 전압 범위입니다.

7. *numInputs*
입력 포트의 개수입니다.

8. *numSupervised*
supervised 입력 포트의 개수입니다.

9. *portIndex*
입력 포트 번호입니다.

10. *enabled*
supervised input으로 사용하는지 결정하는 flag입니다.

11. *supervised_index*
supervised input의 저항값 유형입니다.

값	설명
0	1k 저항
1	2.2k 저항
2	4.7k 저항
3	10k 저항
255	사용자 정의

12. *aux*
[+2.9.8] Aux의 동작을 설정합니다.

비트 위치	비트 수	멤버명	설명
0	4	tamperAuxIndex	Tamper로 사용할 Aux 번호 (0:None, 1:Aux0, 2:Aux1)
7	4	acFailAuxIndex	AC Fail로 사용할 Aux 번호 (0:None, 1:Aux0, 2:Aux1)
8	1	aux0Type	Aux 0의 타입 (0: NO, 1:NC)
9	1	aux1Type	Aux 1의 타입 (0: NO, 1:NC)
10	6	reserved	예약된 공간

13. *config*
supervised 입력 신호 유형을 판단하는 설정값으로, supervised input의 저항값이 사용자 정의일 경우에만 의미가 있습니다.

BS2WlanConfig

```
typedef struct {
    uint8_t enabled;
    uint8_t operationMode;
    uint8_t authType;
    uint8_t encryptionType;
    char essid[BS2_WLAN_SSID_SIZE];
    char authKey[BS2_WLAN_KEY_SIZE];
    uint8_t reserved2[32];
} BS2WlanConfig;
```

1. *enabled*
무선랜 사용 유무입니다.

2. *operationMode*
무선랜 사용 유형입니다.

값	설명
0	infrastructure
1	Ad-hoc

3. *authType*
무선랜 인증 유형입니다.

값	설명
0	Open
1	Shared
2	WPA-PSK
3	WPA2-PSK

4. *encryptionType*
무선랜 암호화 방식입니다.

값	설명
0	없음
1	WEP
2	TKIP/AES
3	AES
3	TKIP

5. *essid*
무선랜 네트워크 식별자입니다.

6. *authKey*
무선랜 네트워크 비밀번호입니다.

7. *reserved*
예약된 공간입니다.

BS2Trigger

```
typedef struct {
    uint16_t code;
    uint8_t reserved[2];
} BS2EventTrigger;

typedef struct {
    uint8_t port;
    uint8_t switchType;
    uint16_t duration;
    uint32_t scheduleID;
} BS2InputTrigger;

typedef struct {
    uint32_t type;
    uint32_t scheduleID;
} BS2ScheduleTrigger;

typedef struct {
    uint32_t deviceID;
    uint8_t type;
    uint8_t reserved;
    uint16_t ignoreSignalTime;

    union {
        BS2EventTrigger event;
        BS2InputTrigger input;
        BS2ScheduleTrigger schedule;
    }
} BS2Trigger;
```

1. *code*
trigger를 발생시킬 event log입니다.

2. *reserved*
예약된 공간입니다.

3. *port*
trigger를 발동시킬 입력 포트 번호입니다.

4. *switchType*
입력 포트의 동작 모드로 대기 상태가 'off'이고 입력 신호가 on일 때 trigger를 발동시킵니다.

값	설명
0	Normally Open
1	Normally Closed

5. *duration*
trigger를 발동시키기 위한 입력 신호의 지속 시간으로 단위는 밀리초(ms)이며, 최소값은 100입니다.

6. *scheduleID*
trigger를 동작하기 위한 타임 스케줄입니다.

7. *type*
schedule trigger의 유형입니다.

값	설명
0	시작 schedule trigger
1	끝 schedule trigger

8. *scheduleID*
trigger를 동작하기 위한 타임 스케줄입니다.

9. *deviceID*
trigger를 수행할 장치 식별자입니다.

10. *type*
trigger의 유형입니다.

값	설명
0	None
1	Event trigger
2	Input trigger
3	Schedule trigger

11. *reserved*
예약된 공간입니다.

12. *ignoreSignalTime*
[+2.9.6] 밀리초 단위로 지정하며 일정 시간 내 동일한 입력 신호가 반복적으로 입력되는 것을 무시하기 위한 설정입니다.
슈프리마 장치가 타사 컨트롤러와 wiegand로 연결된 환경에서
특정 모델의 컨트롤러로부터 슈프리마 장치로 인증에 대한 피드백이 반복적으로 입력되는 경우
이 설정을 통해 해당 시간만큼 강제로 입력 신호가 무시되도록 지정할 수 있습니다.

BS2Action

```
typedef struct {
    uint32_t signalID;
    uint16_t count;
    uint16_t onDuration;
    uint16_t offDuration;
    uint16_t delay;
} BS2Signal;

typedef struct {
    uint8_t portIndex;
    uint8_t reserved[3];
    BS2Signal signal;
} BS2OutputPortAction;

typedef struct {
    uint8_t relayIndex;
    uint8_t reserved[3];
    BS2Signal signal;
} BS2RelayAction;

typedef struct {
    uint8_t color;
    uint8_t reserved[1];
    uint16_t duration;
    uint16_t delay;
} BS2LedSignal;

typedef struct {
    uint16_t count;
    uint8_t reserved[2];
    BS2LedSignal signal[3];
} BS2LedAction;

typedef struct {
    uint8_t tone;
    uint8_t fadeout;
    uint16_t duration;
    uint16_t delay;
} BS2BuzzerSignal;

typedef struct {
    uint16_t count;
    uint8_t reserved[2];
    BS2BuzzerSignal signal[3];
} BS2BuzzerAction;

typedef struct {
    uint8_t duration;
    uint8_t reserved[3];
    uint32_t displayID;
    uint32_t resourceID;
} BS2DisplayAction;

typedef struct {
    uint8_t count;
    uint16_t soundIndex;
    uint8_t reserved[5];
} BS2SoundAction;

typedef struct {
    uint32_t deviceID;
    uint8_t type;
    uint8_t stopFlag;
    uint16_t delay;
    union {
        BS2RelayAction relay;
        BS2OutputPortAction outputPort;
        BS2DisplayAction display;
        BS2SoundAction sound;
        BS2LedAction led;
        BS2BuzzerAction buzzer;
    };
} BS2Action;
```

1. *signalID*
애플리케이션에서 신호 유형을 관리하고자 할 때 사용하는 인덱스입니다.

2. *count*
신호 수행 횟수입니다.

3. *onDuration*
on 신호의 지속 시간으로 단위는 밀리초(ms)입니다.

4. *offDuration*
off 신호의 지속 시간으로 단위는 밀리초(ms)입니다.

5. *delay*
신호를 발생시키기 전에 대기하는 시간으로 단위는 밀리초(ms)입니다. 예를 들어, count(2), onDuration(100), offDuration(100), delay(50)이라면 다음과 같이 신호를 발생시킵니다.

50ms 대기	signal on(100)	signal off(100)	signal on(100)	signal off(100)

6. *portIndex*
TTL 출력 포트 번호입니다.

7. *reserved*
예약된 공간입니다.

8. *relayIndex*
Relay 출력 포트 번호입니다.

9. *reserved*
예약된 공간입니다.

10. *color*
LED 색상 종류입니다.

값	설명
0	LED Off
1	빨강 LED
2	노랑 LED
3	초록 LED
4	청록 LED
5	파랑 LED
6	자홍 LED
7	백색 LED

11. *reserved*
예약된 공간입니다.

12. *duration*
LED 점멸 지속 시간으로 단위는 밀리초(ms)입니다.

13. *delay*
LED 점멸하기 전에 대기하는 시간으로 단위는 밀리초(ms)입니다.

14. *count*
LED 신호 횟수로 0일 경우 비활성화되고 -1일 경우 무한 반복됩니다.

15. *reserved*
예약된 공간입니다.

16. *tone*
Buzzer의 소리 세기(톤)입니다.

값	설명
0	진동 없음
1	약한 진동
2	중간 진동
3	강한 진동

17. *count*
Buzzer 신호 횟수로 0일 경우 비활성화되고 -1일 경우 무한 반복됩니다.

18. *reserved*
예약된 공간입니다.

19. *duration*
Display 동작의 지속 시간으로 단위는 밀리초(ms)입니다.

20. *reserved*
Display 동작의 지속 시간으로 단위는 밀리초(ms)입니다.

21. *displayID*
추후 지원 예정입니다.

22. *resourceID*
추후 지원 예정입니다.

23. *count*
Sound 신호 횟수입니다.

24. *soundIndex*
Sound resource 인덱스입니다.

값	설명
0	Welcome sound
1	Auth success sound
2	Auth fail sound

25. *deviceID*
Action을 수행하는 장치 식별자 입니다.

26. *type*
Action 유형입니다.

[DoorModule-20, CoreStation-40]
Action type이 relay 또는 TTL(Output)이면서, action 장치가 DM20, CS40 이라면, action type은 relay action (6) 으로만 설정하여야 합니다. (TTL 설정불가)

[DM20]

- Action type : Relay
- relay.relayIndex : 0 ~ 3 (RELAY 0 ~ 3)
- relay.relayIndex : 4 ~ 9 (OUTPUT 0 ~ 5)

[CS40]

- Action type : Relay
- relay.relayIndex : 0 ~ 3 (RELAY 0 ~ 3)
- relay.relayIndex : 4 ~ 11 (OUTPUT 0 ~ 7)

값	설명
0	None
1	Lock device
2	Unlock device
3	Reboot device
4	Release alarm
5	General input
6	Relay action

값	설명
7	TTL action
8	Sound action
9	Display action
10	Buzzer action
11	Led action
12	Fire alarm input
13	Auth Success(Access granted)
14	Auth Fail(Access denied)
15	Lift action

27. *stopFlag*
Action을 중지하는 조건을 명시합니다.
만일 이 값이 1로 설정되고 door sensor를 통해 신호가 감지되면 action을 멈추게 됩니다.
만일 이 값이 2로 설정되면 현재 action정보에 국한하여 중단할 수 있습니다.
일반적으로 action을 중지하는 관련 API들은 구역id를 가지고 호출하게되는데, 이 경우 구역내 모든 장치가 action을 멈추게 됩니다.
stopFlag를 2로 설정하여 action정보를 내려주게되면 해당 장치의 경보만 선별적으로 제어할 수 있습니다.

값	설명
0	중지 안함
1	출입문 닫힘 시 중지
2	명령을 통해 중지 (V2.6.0 추가)

28. *delay*
Action 발생에대한 대기시간을 명시합니다. 단위는 밀리초(ms)입니다.

BS2TriggerActionConfig

```
typedef struct {
    uint8_t numItems;
    uint8_t reserved[3];
    BS2TriggerAction items[BS2_MAX_TRIGGER_ACTION];
    uint8_t reserved2[32];
} BS2TriggerActionConfig;
```

1. *numItems*
trigger action의 개수입니다.

2. *reserved*
예약된 공간입니다.

3. *items*
trigger action을 나열한 리스트로 최대 128개까지 설정할 수 있습니다.

4. *reserved2*
예약된 공간입니다.

BS2EventConfig

```
typedef struct {
    uint32_t numImageEventFilter;
    struct {
        uint8_t mainEventCode;
        uint8_t reserved[3];
        uint32_t scheduleID;
    } imageEventFilter[BS2_EVENT_MAX_IMAGE_CODE_COUNT];
    uint8_t reserved[32];
} BS2EventConfig;
```

1. *numImageEventFilter*
image log filter의 개수입니다.

2. *mainEventCode*
image log를 남길 log main code입니다.

3. *reserved*
예약된 공간입니다.

4. *scheduleID*
image log를 저장해야 하는 타임 스케줄입니다.

5. *reserved*
예약된 공간입니다.

BS2WiegandMultiConfig

```
typedef struct {
    uint32_t formatID;
    BS2WiegandFormat format;
    uint8_t reserved[32];
} BS2WiegandInConfig;

typedef struct {
    BS2WiegandInConfig formats[MAX_WIEGAND_IN_COUNT];
    uint8_t reserved[32];
} BS2WiegandMultiConfig;
```

1. *formatID*
WiegandFormat Index입니다.

2. *format*
WiegandFormat 구조체입니다.

3. *reserved*
예약된 공간입니다.

4. *formats*

WiegandInConfig 구조체를 15개를 가질 수 있습니다.

5. *reserved*
예약된 공간입니다.

BS1CardConfig

```
typedef struct {
    enum {
        MIFARE_KEY_SIZE = 6,
        MIFARE_MAX_TEMPLATE = 4,

        VALID_MAGIC_NO = 0x1f1f1f1f,
    };

    // Options
    uint32_t     magicNo;
    uint32_t     disabled;
    uint32_t     useCSNOnly;         // default 0
    uint32_t     bioentryCompatible; // default 0

    // Keys
    uint32_t    useSecondaryKey;
    uint32_t     reserved1;
    uint8_t     primaryKey[MIFARE_KEY_SIZE];
    uint8_t        reserved2[2];
    uint8_t     secondaryKey[MIFARE_KEY_SIZE];
    uint8_t        reserved3[2];

    // Layout
    uint32_t     cisIndex;
    uint32_t     numOfTemplate;
    uint32_t     templateSize;
    uint32_t     templateStartBlock[MIFARE_MAX_TEMPLATE];

    uint32_t     reserve4[15];

} BS1CardConfig;
```

1. *magicNo*
사용하지 않는 변수입니다

2. *disabled*
비활성화 상태인지를 나타내는 flag 값입니다.

3. *useCSNOnly*
CSN만 사용하는 여부입니다.

4. *bioentryCompatible*
boientry 호환 정보입니다.

5. *useSecondaryKey*
사용하지 않는 변수입니다.

6. *reserved1*
예약된 공간입니다.

7. *primaryKey*
사용하지 않는 변수입니다.

8. *reserved2*
예약된 공간입니다.

9. *secondaryKey*
사용하지 않는 변수입니다.

10. *reserved3*
예약된 공간입니다.

11. *cisIndex*
cis 식별자입니다.

12. *numOfTemplate*
템플릿 개수입니다.

13. *templateSize*
템플릿 크기입니다.

14. *templateStartBlock*
data storage에서의 start block index입니다.

15. *reserved4*
예약된 공간입니다.

BS2SystemConfigExt

```
typedef struct {
    uint8_t primarySecureKey[SEC_KEY_SIZE];
    uint8_t secondarySecureKey[SEC_KEY_SIZE];

    uint8_t reserved3[32];
} BS2SystemConfigExt;
```

1. *primarySecureKey*
Master-Slave 장치간의 첫번째 암호화 키입니다.

2. *secondarySecureKey*
cMaster-Slave 장치간의 두번째 암호화 키입니다.

3. *reserved3*
예약된 공간입니다.

BS2VoipConfig

```
typedef struct {
    BS2_URL            serverUrl;            ///
    BS2_PORT        serverPort;           ///
    BS2_USER_ID        userID;               ///
    BS2_USER_ID        userPW;               ///

    uint8_t            exitButton;           /// << *, #, 0~9
    uint8_t            dtmfMode;           ///
    BS2_BOOL        bUse;               ///
    uint8_t            reseverd[1];        ///

        uint32_t                numPhonBook;
    BS2UserPhoneItem     phonebook[BS2_VOIP_MAX_PHONEBOOK];  ///

    uint8_t                 reserved2[32];        ///
} BS2VoipConfig;
```

1. *serverUrl*
BioStar 애플리케이션 서버 URL이며, 최대 256자까지 입력할 수 있습니다

2. *serverPort*
connectionMode가 server mode일 때 사용되며, BioStar 애플리케이션의 포트 번호입니다.

3. *userID*
사용자 식별자입니다.

4. *userPW*
사용자 비밀번호입니다.

5. *exitButton*
출구 버튼 기호입니다. (*, #, 0~9)

값	설명
0	*
1	#
2 ~ 11	0 ~ 9

6. *dtmfMode*
전화기를 들고 번호판을 누를때 들리는 음을 말합니다.

7. *bUse*
사용여부입니다.

8. *reseverd*
예약된 공간입니다.

9. *numPhonBook*
전호번호부 개수입니다.

10. *phonebook*
전화번호부를 나열한 리스트로 최대 32까지 설정할 수 있습니다.

8. *reserved2*
예약된 공간입니다.

BS2FaceConfig

```
typedef struct {
    uint8_t       securityLevel;
    uint8_t       lightCondition;
    uint8_t       enrollThreshold;
    uint8_t       detectSensitivity;

    uint16_t      enrollTimeout;
    uint8_t       lfdLevel;
    bool          quickEnrollment;

    uint8_t       previewOption;
    bool          checkDuplicate;
    uint8_t       operationMode;
    uint8_t       maxRotation;

    // Deprecated
    struct {
        uint16_t  min;
        uint16_t  max;
    } faceWidth;

    // Deprecated
    struct {
        uint16_t  x;
        uint16_t  width;
    } searchRange;

    struct {
        uint8_t min;       // 30 ~ 100
        uint8_t max;       // 40 ~ 100, 255
    } detectDistance;         ////< 2 bytes

    BS2_BOOL wideSearch;   ///< 1 byte
    uint8_t unused;

    uint8_t unableToSaveImageOfVisualFace;
    uint8_t reserved[13];
} BS2FaceConfig;
```

1. *securityLevel*
시스템 전반적으로 사용할 얼굴 인증 보안 레벨입니다.

값	설명
0	기본
1	보안 레벨 높음
2	보안 레벨 매우 높음

2. *lightCondition*
빛 조건입니다.

값	설명
0	옥내
1	옥외
2	자동
3	[+ 2.8] 사용 안함

3. *enrollThreshold*
얼굴 등록의 임계값입니다. 등록시 어느 정도 포즈 움직임을 허용하는지에 대한 값입니다.

값	설명
0	THRESHOLD_0 (엄격)
1	THRESHOLD_1
2	THRESHOLD_2
3	THRESHOLD_3
4	THRESHOLD_4 (기본)
5	THRESHOLD_5
6	THRESHOLD_6
7	THRESHOLD_7
8	THRESHOLD_8
9	THRESHOLD_9 (느슨)

4. *detectSensitivity*
얼굴 센서의 민감도 설정입니다.

값	설명
0	꺼짐
1	낮음
2	기본
3	높음

5. *enrollTimeout*
`IR Face` 기반 : 얼굴 스캔 대기 시간으로 기본값은 60초입니다.

값	설명
BS2_FACE_ENROLL_TIMEOUT_MIN	30
BS2_FACE_ENROLL_TIMEOUT_MAX	60
BS2_FACE_ENROLL_TIMEOUT_DEFAULT	BS2_FACE_ENROLL_TIMEOUT_MAX

`Visual Face` 기반 : [+ V2.7.1] 얼굴 스캔 대기 시간으로 기본값은 20초입니다.

값	설명
BS2_FACE_EX_ENROLL_TIMEOUT_MIN	10

값	설명
BS2_FACE_EX_ENROLL_TIMEOUT_MAX	20
BS2_FACE_EX_ENROLL_TIMEOUT_DEFAULT	BS2_FACE_EX_ENROLL_TIMEOUT_MAX

6. *lfdLevel*
[+ 2.6.3] 위조된 얼굴을 여과할 수 있는 민감도를 설정할 수 있습니다.
`IR Face` 기반 : 기본값은 0입니다.
`Visual Face` 기반 : [+ 2.7.1] 기본값이 1입니다.

값	설명
0	사용안함
1	민감도 낮음
2	민감도 보통
3	민감도 높음

7. *quickEnrollment*
[+ 2.6.3] 빠른 얼굴 등록 사용 여부를 설정할 수 있습니다.
이 값을 true로 설정하면 얼굴 등록 절차가 1단계로 설정되며, false로 설정할 경우 3단계로 설정됩니다.
고품질의 얼굴 템플릿을 등록하려면 false를 설정하십시오.

8. *previewOption*
[+ 2.6.3] IR 인증 가이드를 의미하며, 얼굴 인증 시 preview 화면 출력 설정을 변경할 수 있습니다.
`FaceLite`에서만 사용됩니다.

값	설명
0	Preview 표시 안함
1	인증 시작 시는 preview를 표시하지 않으며, 1/2 지점에 표시
2	인증 시도 전구간에 걸쳐 preview를 표시

9. *checkDuplicate*
[+ 2.6.4] true로 설정하면 중복된 얼굴 정보인지 판단합니다.

10. *operationMode*
[+ 2.7.1] `FaceStation F2 V1.0.1`미만은 아래의 동작 모드로 설정가능하며, 기본값은 Fusion 모드입니다.

값	모드	설명	기본값
0	Fusion 모드	Visual matching + IR matching	기본
1	Visual 모드	Visual matching	
2	Visual + IR검출	Visual matching, IR은 얼굴 검출만	

`FaceStation F2 V1.0.1`이상, `Visual Face` 기반는 아래와 같은 의미로 사용됩니다.

값	모드	설명	기본값
0	Fusion 모드	Visual matching + IR matching	기본
1	Fast 모드	Visual matching	

11. *maxRotation*
[+ 2.7.1] `Visual Face` 기반 일반적인 상황에서 얼굴을 검출할 때 얼굴의 방향은 정면일 것입니다.
이 경우 장치는 얼굴 검출 시 정면에서 어느 정도 회전된(왜곡된) 이미지인지 판단할 수 있습니다.
이를 통해 일정 각도 이상 회전된 이미지의 경우 검출 실패 처리를 할 수 있습니다.
maxRotation은 이런 경우의 최대 허용치를 나타내며 기본값은 15도입니다.

[+ 2.9.6] Angle 추가 정의

정의	값
BS2_MAX_ROTATION_DEFAULT	15
BS2_MAX_ROTATION_ANGLE_15	15
BS2_MAX_ROTATION_ANGLE_30	30
BS2_MAX_ROTATION_ANGLE_45	45
BS2_MAX_ROTATION_ANGLE_60	60
BS2_MAX_ROTATION_ANGLE_75	75
BS2_MAX_ROTATION_ANGLE_90	90
BS2_MAX_ROTATION_ANGLE_MAX	90

12. *faceWidth*
[+ 2.7.1] `Visual Face` 기반 얼굴 이미지의 폭을 나타내며 폭의 최소값, 최대값을 지정할 수 있습니다.
최소값, 최대값의 기본값은 아래와 같습니다.

	기본값(min)	기본값(max)
FSF2	66	250
BS3	130	350
BEW3	130	350

13. *searchRange*
[+ 2.7.1] `Visual Face` 기반 얼굴 감지 범위를 나타내며 범위의 x값(가로 좌표)과 x값으로부터의 폭을 지정할 수 있습니다.
x값과 폭의 기본값은 아래와 같습니다.

	기본값(x)	기본값(width)
FSF2	144	432
BS3	90	540
BEW3	90	540

14. *detectDistance*
[+ 2.8.3] `Visual Face` 기반 얼굴 감지 거리의 최소값과 최대값을 설정합니다.
픽셀 단위로 얼굴 위치를 지정하는 faceWidth는 사용하기 난해하므로 더 이상 사용하지 않습니다.
대신 피사체(얼굴)의 감지 거리를 얼마로 할지 만 정합니다. 거리는 cm로 설정하며, 10의 배수로 입력하여야 합니다.

	최소 감지거리의 최소값	최소 감지거리의 최대값	최소 감지거리(기본값)	최대 감지거리의 최소값	최대 감지거리의 최대값	최대 감지거리(무제한)	최대 감지거리(기본값)
FSF2	30	130	30	40	130	255	130
BS3	30	100	30	40	100	255	100
BEW3	30	100	30	40	100	255	100

15. *wideSearch*
[+ 2.8.3] `BioStation 3, BioEntry W3` 얼굴 감지 범위를 넓힙니다.
x좌표와 width를 지정해야하는 난해한 설정인 searchRange는 더 이상 사용하지 않습니다.
대신 얼굴 감지를 표준(false) 설정으로 할지, 광역(true)으로 할지 여부만 지정합니다.
광역으로 감지하는 범위에 대한 세부 설정은 장치 내부 설정 값을 따릅니다. 사용자는 이 값을 변경할 수 없습니다.

이 설정 값이 커지면(true) 카메라를 통해 넓은 범위의 피사체를 찾아내므로, 피사체가 여럿이 되는 경우 의도치 않은 인증으로 연결될 수 있습니다.
기본 설정 값은 false입니다.

16. *unused*
예약된 공간입니다.

17. *unableToSaveImageOfVisualFace*
[+ 2.9.6] Visual face를 크리덴셜로 사용하는 장치에서 얼굴 이미지를 장치에 저장할 것인지를 나타냅니다.
이 설정을 키면 장치 내 모든 사용자의 얼굴 정보 중 이미지 정보가 즉시 삭제되며 템플릿만 남게됩니다.
또한, BS2_EnrollUserFaceEx API를 통해 사용자 이미지가 포함된 얼굴 정보를 가져오더라도 장치는 이를 무시하게됩니다.
기본값은 false 이며, 얼굴과 이미지 모두 저장입니다.

18. *reserved*
예약된 공간입니다.

BS2Rs485ConfigEX

```
typedef struct {
    uint32_t baudRate;
    uint8_t channelIndex;
    uint8_t useRegistance;
    uint8_t numOfDevices;
    uint8_t reserved[1];
    BS2Rs485SlaveDeviceEX slaveDevices[BS2_RS485_MAX_SLAVES_PER_CHANNEL];
} BS2Rs485ChannelEX;

typedef struct {
    uint8_t mode[BS2_RS485_MAX_CHANNELS_EX];
    uint8_t numOfChannels;
    uint8_t reserved[2];
    uint8_t reserved1[32];
    BS2Rs485ChannelEX channels[BS2_RS485_MAX_CHANNELS];
} BS2Rs485ConfigEX;
```

1. *baudRate*
RS485 통신 속도이며 설정할 수 있는 범위는 다음과 같습니다.

값
9600
19200

값
38400
57600
115200

2. *channelIndex*
RS485 network의 통신 채널 번호입니다.

3. *useRegistance*
종단 저항을 설정할지 결정하는 flag입니다.

4. *numOfDevices*
슬레이브 장치의 개수입니다.

5. *slaveDevices*
슬레이브 장치를 나열한 리스트로 최대 32개까지 정의할 수 있습니다.

6. *mode*
RS485 네트워크에서 어떤 모드로 동작할지 결정하는 flag입니다 .

값	설명
1	Master
2	Slave
3	Standalone

CoreStation은 Master로만 사용 가능합니다. 다른 값은 설정할 수 없습니다.

7. *numOfChannels*
RS485 채널의 개수입니다.

8. *reserved*
예약된 공간입니다.

9. *reserved1*
예약된 공간입니다.

10. *channels*
RS485 채널을 나열한 리스트로 최대 8개까지 설정할 수 있습니다.

BS2CardConfigEx

```
typedef struct {
    uint8_t oid_ADF[13];            ///  //유효
값//{0x2A,0x85,0x70,0x81,0x1E,0x10,0x00,0x07,0x00,0x00,0x02,0x00,0x00}
    uint8_t size_ADF;              //
    uint8_t reserved1[2];           ///
    uint8_t oid_DataObjectID[8];
    uint16_t size_DataObject[8];
    uint8_t primaryKeyAuth[16];             //유효값
    uint8_t secondaryKeyAuth[16];       /// //유효값
    uint8_t reserved2[24];
} BS2SeosCard;
typedef struct {
    BS2SeosCard seos;
    uint8_t reserved[24];
} BS2CardConfigEx;
```

1. *oid_ADF*
ADF 주소값을 표시합니다. (변경할 수 없습니다.)

2. *size_ADF*
ADF size을 표시합니다.

3. *reserved1*
예약된 공간입니다.

4. *oid_DataObjectID*
DataObjectID를 표시합니다.

5. *size_DataObject*
DataObject size를 표시합니다.

6. *primaryKeyAuth*
Seoscard 정보에 접근하기 위한 첫번째 암호화 키입니다.

7. *secondaryKeyAuth*
Seoscard 정보에 접근하기 위한 두번째 암호화 키입니다.

8. *reserved2*
예약된 공간입니다.

9. *seos*
BS2SeosCard 정보입니다.

10. *reserved*
예약된 공간입니다.

BS2DstConfig

```
enum {
    BS2_MAX_DST_SCHEDULE = 2,
};

typedef struct {
    uint16_t year;        // year, 0 means every year.
    uint8_t month;        // [0, 11] : months since January
    int8_t ordinal;        // [0, -1] : first, second, ..., last
    uint8_t weekDay;    // [0, 6] : days since Sunday
    uint8_t hour;         // [0, 23]
    uint8_t minute;         // [0, 59]
    uint8_t second;         // [0, 59]
} BS2WeekTime;

typedef struct {
    BS2WeekTime startTime;
    BS2WeekTime endTime;
    int32_t timeOffset;     // in seconds
    uint8_t reserved[4];
} BS2DstSchedule;

typedef struct {
    uint8_t numSchedules;
    uint8_t reserved[31];

    BS2DstSchedule schedules[BS2_MAX_DST_SCHEDULE];
} BS2DstConfig;
```

1. *year*
년을 의미하며, 0으로 설정되면 매년을 의미합니다.

2. *month*
월을 의미하며, 0 부터 11 [1월-12월] 사이의 값을 가집니다.

3. *ordinal*
0부터 시작되며 첫 번째, 두 번째 등의 순서를 의미합니다.

4. *weekDay*
요일을 의미하며, 0은 일요일을, 1은 월요일을 의미합니다.

5. *hour*
24시간 기준으로 시간을 지정합니다.

6. *minute*
분을 지정합니다.

7. *second*
초를 지정합니다.

8. *startTime*
시작일시을 의미합니다.

9. *endTime*
종료일시를 의미합니다.

10. *timeOffset*
DST를 적용할 시간을 초 단위로 적용할 수 있습니다.
예를 들어 1시간을 적용하고 싶은 경우, 3600을 입력합니다.

11. *reserved*
예약된 공간입니다.

12. *numSchedules*

적용할 DST schedule의 개수를 의미합니다.

13. *schedules*
DST schedule이며 최대 2개까지 지정할 수 있습니다.

BS2Configs

```
typedef struct {
    uint32_t configMask;
    BS2FactoryConfig factoryConfig;
    BS2SystemConfig systemConfig;
    BS2AuthConfig authConfig;
    BS2StatusConfig statusConfig;
    BS2DisplayConfig displayConfig;
    BS2IpConfig ipConfig;
    BS2IpConfigExt ipConfigExt;
    BS2TNAConfig tnaConfig;
    BS2CardConfig cardConfig;
    BS2FingerprintConfig fingerprintConfig;
    BS2Rs485Config rs485Config;
    BS2WiegandConfig wiegandConfig;
    BS2WiegandDeviceConfig wiegandDeviceConfig;
    BS2InputConfig inputConfig;
    BS2WlanConfig wlanConfig;
    BS2TriggerActionConfig triggerActionConfig;
    BS2EventConfig eventConfig;
    BS2WiegandMultiConfig wiegandMultiConfig;
    BS1CardConfig card1xConfig;
    BS2SystemConfigExt systemExtConfig;
    BS2VoipConfig voipConfig;
    BS2FaceConfig faceConfig;
} BS2Configs;
```

1. *configMask*
configuration을 가져오거나 설정할 mask값입니다.

값	설명
0x0000	None
0x0001	Factory configuration
0x0002	System configuration
0x0004	TCP/IP configuration
0x0008	RS485 configuration
0x0010	Wireless LAN configuration
0x0020	Authentication configuration
0x0040	Card configuration
0x0080	Fingerprint configuration
0x0100	Face configuration
0x0200	Trigger Action configuration

값	설명
0x0400	Display configuration
0x0800	Sound configuration
0x1000	Status Signal(LED, Buzzer) configuration
0x2000	Wiegand configuration
0x4000	USB configuration
0x8000	Time and Attendance configuration
0x10000	Videophone configuration
0x20000	Interphone configuration
0x40000	Voice over IP configuration
0x80000	Input(Supervised input) configuration
0x100000	Wiegand IO Device configuration
0x200000	Time and Attendance configuration
0x400000	DNS and Server url configuration
0x800000	Event configuration
0x1000000	1x Card configuration
0x2000000	Multi-Wiegand configuration
0x4000000	Extended System configuration
~~0x8000000~~	~~Daylight Saving configuration (Deprecated)~~
0x10000000	RS485 Extended configuration
0x20000000	Extended Card configuration
0x40000000	Daylight Saving configuration
0xFFFFFFFF	All configuration

BS2IPV6Config

```
enum {
    BS2_MAX_IPV6_ALLOCATED_ADDR = 8,
};

typedef struct {
    uint8_t useIPV6;
    uint8_t reserved1;
    uint8_t useDhcpV6;
    uint8_t useDnsV6;
    uint8_t reserved[1];
    char staticIpAddressV6[BS2_IPV6_ADDR_SIZE];
    char staticGatewayV6[BS2_IPV6_ADDR_SIZE];
    char dnsAddrV6[BS2_IPV6_ADDR_SIZE];
    char serverIpAddressV6[BS2_IPV6_ADDR_SIZE];
    uint16_t serverPortV6;
    uint16_t sslServerPortV6;
    uint16_t portV6;
    uint8_t numOfAllocatedAddressV6;
    uint8_t numOfAllocatedGatewayV6;
    uint8_t reserved[8];
    char
allocatedIpAddressV6[BS2_IPV6_ADDR_SIZE][BS2_MAX_IPV6_ALLOCATED_ADDR];
    char
allocatedGatewayV6[BS2_IPV6_ADDR_SIZE][BS2_MAX_IPV6_ALLOCATED_ADDR];
} BS2IpConfig;
```

1. *useIPV6*
IP V6 사용여부를 나타내는 flag입니다.

2. *reserved1*
예약된 공간입니다.

3. *useDhcpV6*
DHCP 사용 유무를 나타내는 flag입니다.

4. *useDnsV6*
server addresss를 사용할지 server URL을 사용할 지 결정하는 flag입니다.

5. *staticIpAddressV6*
현재 장치의 정적 IP V6 주소입니다.

6. *staticGatewayV6*
현재 장치의 정적 IP V6 게이트웨이 주소입니다.

7. *dnsAddrV6*
DNS V6 주소입니다.

8. *serverIpAddressV6*
connectionMode가 server mode일 때 사용되며, BioStar 애플리케이션의 IP V6 주소입니다.

9. *serverPortV6*
connectionMode가 server mode일 때 사용되며, BioStar 애플리케이션의 포트 번호입니다.

10. *sslServerPortV6*
connectionMode가 server mode일 때 사용되며, ssl로 연결될 애플리케이션의 서버 포트 번호입니다.

11. *portV6*
현재 장치의 IP V6 포트 번호입니다.

12. *numOfAllocatedAddressV6*
현재 장치에 할당되어진 IP V6 주소의 갯수를 나타냅니다.

13. *numOfAllocatedGatewayV6*
현재 장치에 할당되어진 IP V6 게이트웨이 주소의 갯수를 나타냅니다.

14. *reserved*
예약된 공간입니다.

15. *allocatedIpAddressV6*
현재 장치에 할당되어진 IP V6 주소입니다. numOfAllocatedAddressV6는 할당되어진 주소의 갯수를 나타냅니다.

16. *allocatedGatewayV6*

현재 장치에 할당되어진 IP V6 게이트웨이 주소입니다. numOfAllocatedGatewayV6는 할당되어진 게이트웨이 주소의 갯수를 나타냅니다.

BS2DesFireCardConfigEx

```
typedef struct {
    uint8_t appMasterKey[16];
    uint8_t fileReadKey[16];
    uint8_t fileWriteKey[16];
    uint8_t fileReadKeyNumber;
    uint8_t fileWriteKeyNumber;
    uint8_t reserved[2];
} BS2DesFireAppLevelKey;                          ///< 52 bytes

typedef struct {
    BS2DesFireAppLevelKey desfireAppKey;     ///< 52 bytes
    uint8_t reserved[16];
} BS2DesFireCardConfigEx;                         ///< 68 bytes
```

1. *appMasterKey*
DesFire의 application master key 값 입니다.

2. *fileReadKey*
파일 읽기에 사용되는 key 값 입니다.

3. *fileWriteKey*
파일 쓰기에 사용되는 key 값 입니다.

4. *fileReadKeyNumber*
파일 읽기용 key의 key index 입니다.

5. *fileWriteKeyNumber*
파일 쓰기용 key의 key index 입니다.

6. *reserved*
예약된 공간입니다.

7. *desfireAppKey*
DesFire의 키 정보를 갖는 구조체입니다.

8. *reserved*
예약된 공간입니다.

BS2AuthConfigExt

```
typedef struct {
    uint32_t extAuthSchedule[BS2_MAX_NUM_OF_EXT_AUTH_MODE];
    uint8_t useGlobalAPB;
    uint8_t globalAPBFailAction;
    uint8_t useGroupMatching;
    uint8_t reserved;

    uint8_t reserved2[4];

    uint8_t usePrivateAuth;
    uint8_t faceDetectionLevel;
    uint8_t useServerMatching;
    uint8_t useFullAccess;

    uint8_t matchTimeout;
    uint8_t authTimeout;
    uint8_t numOperators;
    uint8_t reserved3[1];

    struct {
        char userID[BS2_USER_ID_SIZE];
        uint8_t level;
        uint8_t reserved[3];
    } operators[BS2_MAX_OPERATORS];

    uint8_t reserved4[256];
} BS2AuthConfigExt;
```

1. *extAuthSchedule*
각 인증 모드가 활성화될 때 실행할 일정 식별자입니다.
배열의 순서대로 아래의 의미를 가지며,
배열 내의 값이 0보다 큰경우 해당 인증모드는 활성화 되어 있습니다.
아래 설명 중 생체정보는 장치에 따라 지문 또는 얼굴을 의미합니다.

순서	의미	설명
11	BS2_EXT_AUTH_MODE_FACE_ONLY	얼굴
12	BS2_EXT_AUTH_MODE_FACE_FINGERPRINT	얼굴 + 지문
13	BS2_EXT_AUTH_MODE_FACE_PIN	얼굴 + PIN
14	BS2_EXT_AUTH_MODE_FACE_FINGERPRINT_OR_PIN	얼굴 + 지문/PIN
15	BS2_EXT_AUTH_MODE_FACE_FINGERPRINT_PIN	얼굴 + 지문 + PIN
16	BS2_EXT_AUTH_MODE_FINGERPRINT_ONLY	지문
17	BS2_EXT_AUTH_MODE_FINGERPRINT_FACE	지문 + 얼굴
18	BS2_EXT_AUTH_MODE_FINGERPRINT_PIN	지문 + PIN
19	BS2_EXT_AUTH_MODE_FINGERPRINT_FACE_OR_PIN	지문 + 얼굴/PIN
20	BS2_EXT_AUTH_MODE_FINGERPRINT_FACE_PIN	지문 + 얼굴 + PIN
21	BS2_EXT_AUTH_MODE_CARD_ONLY	카드
22	BS2_EXT_AUTH_MODE_CARD_FACE	카드 + 얼굴
23	BS2_EXT_AUTH_MODE_CARD_FINGERPRINT	카드 + 지문
24	BS2_EXT_AUTH_MODE_CARD_PIN	카드 + PIN
25	BS2_EXT_AUTH_MODE_CARD_FACE_OR_FINGERPRINT	카드 + 얼굴/지문

순서	의미	설명
26	BS2_EXT_AUTH_MODE_CARD_FACE_OR_PIN	카드 + 얼굴/PIN
27	BS2_EXT_AUTH_MODE_CARD_FINGERPRINT_OR_PIN	카드 + 지문/PIN
28	BS2_EXT_AUTH_MODE_CARD_FACE_OR_FINGERPRINT_OR_PIN	카드 + 얼굴/지문/PIN
29	BS2_EXT_AUTH_MODE_CARD_FACE_FINGERPRINT	카드 + 얼굴 + 지문
30	BS2_EXT_AUTH_MODE_CARD_FACE_PIN	카드 + 얼굴 + PIN
31	BS2_EXT_AUTH_MODE_CARD_FINGERPRINT_FACE	카드 + 지문 + 얼굴
32	BS2_EXT_AUTH_MODE_CARD_FINGERPRINT_PIN	카드 + 지문 + PIN
33	BS2_EXT_AUTH_MODE_CARD_FACE_OR_FINGERPRINT_PIN	카드 + 얼굴/지문 + PIN
34	BS2_EXT_AUTH_MODE_CARD_FACE_FINGERPRINT_OR_PIN	카드 + 얼굴 + 지문/PIN
35	BS2_EXT_AUTH_MODE_CARD_FINGERPRINT_FACE_OR_PIN	카드 + 지문 + 얼굴/PIN
36	BS2_EXT_AUTH_MODE_ID_FACE	ID + 얼굴
37	BS2_EXT_AUTH_MODE_ID_FINGERPRINT	ID + 지문
38	BS2_EXT_AUTH_MODE_ID_PIN	ID + PIN
39	BS2_EXT_AUTH_MODE_ID_FACE_OR_FINGERPRINT	ID + 얼굴/지문
40	BS2_EXT_AUTH_MODE_ID_FACE_OR_PIN	ID + 얼굴/PIN
41	BS2_EXT_AUTH_MODE_ID_FINGERPRINT_OR_PIN	ID + 지문/PIN
42	BS2_EXT_AUTH_MODE_ID_FACE_OR_FINGERPRINT_OR_PIN	ID + 얼굴/지문/PIN
43	BS2_EXT_AUTH_MODE_ID_FACE_FINGERPRINT	ID + 얼굴 + 지문
44	BS2_EXT_AUTH_MODE_ID_FACE_PIN	ID + 얼굴 + PIN
45	BS2_EXT_AUTH_MODE_ID_FINGERPRINT_FACE	ID + 지문 + 얼굴
46	BS2_EXT_AUTH_MODE_ID_FINGERPRINT_PIN	ID + 지문 + PIN
47	BS2_EXT_AUTH_MODE_ID_FACE_OR_FINGERPRINT_PIN	ID + 얼굴/지문 + PIN
48	BS2_EXT_AUTH_MODE_ID_FACE_FINGERPRINT_OR_PIN	ID + 얼굴 + 지문/PIN
49	BS2_EXT_AUTH_MODE_ID_FINGERPRINT_FACE_OR_PIN	ID + 지문 + 얼굴/PIN

2. *useGlobalAPB*
광역 이중 인증 방지 구역을 활성화할지 결정하는 flag입니다.

3. *globalAPBFailAction*
광역 이중 인증 위반인지 BioStar 애플리케이션에 질의를 할 수 없거나 응답이 없을때 기본적으로 수행해야 할 동작입니다.

값	설명
0	APB 검사 안함
1	Soft APB로 동작
2	Hard APB로 동작

4. *useGroupMatching*
얼굴 그룹 매칭을 사용할지 결정하는 flag입니다.

5. *reserved*
예약된 공간입니다.

6. *reserved2*
예약된 공간입니다.

7. *usePrivateAuth*
개별 인증 모드를 사용할지 결정하는 flag입니다.

8. *faceDetectionLevel*
A2에서 사용자를 인증할 때 얼굴 검출 레벨값이며, 지정한 레벨보다 낮은 레벨로 얼굴이 검출되면 인증 실패로 처리됩니다.
설정하면 Normal/Strict에 따른 카메라 뷰가 표시되며, 인증 성공 시의 이미지 로그가 얼굴로 인식되지 않으면 접근이 거부됩니다. 기본값은 0입니다.

값	설명
0	얼굴 검출을 하지 않음
1	Normal mode
2	Strict mode

A2에서만 설정 가능하며, FaceStation2나 FaceLite에서는 사용되지 않습니다.

9. *useServerMatching*
지문 인증이나 얼굴 인식을 Matching server에서 수행할지 결정하는 flag입니다.

10. *useFullAccess*
사용하지 않는 변수입니다.

11. *matchTimeout*
지문 인증이나 얼굴 인식에서 최대 응답 시간이며 단위는 초(sec)를 사용합니다.

12. *authTimeout*
사용자 인증의 최대 응답 시간이며 단위는 초(sec)를 사용합니다.

13. *numOperators*
사용자 권한을 정의하는 operator의 개수입니다.

14. *reserved3*
예약된 공간입니다.

15. *userID*
사용자 식별자입니다.

16. *level*
사용자가 인증되었을 때 식별자에 해당하는 권한을 지정합니다.

값	설명
0	권한 없음
1	관리자 권한
2	시스템 구성을 변경할 수 있는 권한
3	사용자 정보를 변경할 수 있는 권한

주의
Operator 추가 시, 추가하고자 하는 operator의 수를 ***numOperators*** 필드의 값으로 지정해야 합니다.

17. *reserved*
예약된 공간입니다.
18. *reserved4*
예약된 공간입니다.

BS2FaceConfigExt

```
typedef struct {
    uint8_t thermalCheckMode;
    uint8_t maskCheckMode;
    uint8_t reserved[2];

    uint8_t thermalFormat;
    uint8_t reserved2;

    uint16_t thermalThresholdLow;
    uint16_t thermalThresholdHigh;
    uint8_t maskDetectionLevel;
    uint8_t auditTemperature;

    uint8_t useRejectSound;
    uint8_t useOverlapThermal;
    uint8_t useDynamicROI;
    uint8_t faceCheckOrder;
} BS2FaceConfigExt;
```

1. *thermalCheckMode*
열화상 감지 모드를 설정합니다.
HARD로 설정 시, 고열의 기준인 thermalThreshold를 초과하면, 출입에 영향을 주게됩니다.
SOFT로 설정 시, 고열의 기준인 thermalThreshold를 초과하면, 근거를 로그를 남기고 출입에 영향을 주지 않습니다.
thermalCheckMode가 사용안함(0)으로 설정되면,
thermalFormat, thermalThreshold, auditTemperature, useOverlapThermal 의 설정은 무시됩니다.
그리고 useRejectSound의 열화상 감지로 인한 sound발생이, faceCheckOrder의 발열 측정이 무시됩니다.

값	설명	기본값
0	열화상 감지 사용 안함	기본값
1	열화상 감지 사용 (HARD)	
2	열화상 감지 사용 (SOFT)	

2. *maskCheckMode*
`Visual Face` 기반 마스크 착용 감지 모드를 설정합니다.
HARD로 설정 시 maskDetectionLevel에 따라 마스크 착용 위반이 감지될 경우 출입에 영향을 주게됩니다.
SOFT로 설정 시 maskDetectionLevel에 따라 마스크 착용 위반이 감지될 경우 로그가 기록되고 출입에는 영향을 주지 않습니다.
maskCheckMode가 사용안함(0)으로 설정되면, maskDetectionLevel의 설정은 무시됩니다.
또한 useRejectSound의 마스크 착용으로 인한 소리 발생과 faceCheckOrder의 마스크 착용 감지가 무시됩니다.
[+2.9.8] 마스크 착용 금지(3)는 Mask Check Mode가 Check Before Authentication 인 경우만 설정 가능하며, maskDetectionLevel에 따라 마스크를 착용한 사용자는 인증이 실패됩니다. 이 옵션은 BS2DeviceCapabilities - authDenyMaskSupported를 지원하는 장치만 사용할 수 있습니다.

값	설명	기본값
0	마스크 착용 감지 사용 안함	기본값
1	마스크 착용 감지 사용 (HARD)	
2	마스크 착용 감지 사용 (SOFT)	
3	마스크 착용 금지	

3. *reserved*
예약된 공간입니다.

4. *thermalFormat*
온도 단위를 나타내며, 화면상의 온도 표시를 화씨 또는 섭씨로 나타내도록 선택할 수 있습니다.

값	설명	기본값
0	화씨	
1	섭씨	기본값

5. *reserved2*
예약된 공간입니다.

6. *thermalThresholdLow*
고열 판단 기준의 범위값이며, 설정 할 온도의 100을 곱한 값으로 입력하여야 합니다.
또한 섭씨 기준으로만 입력할 수 있습니다.
이 값은 인증 거부의 근거가 되며, 설정 범위는 섭씨로 100 (1º)에서 4500 (45º) 사이 입니다.
기본 값은 3200 (32º)이며, 설정 범위보다 크거나 작은 값을 입력 시 기본 값 3200 (32º)으로 설정됩니다.
그리고 thermalThresholdHigh보다 작은 값을 설정 하여야 합니다.

7. *thermalThresholdHigh*
고열 판단 기준의 범위값이며, 설정 할 온도의 100을 곱한 값으로 입력하여야 합니다.
또한 섭씨 기준으로만 입력할 수 있습니다.
이 값은 인증 거부의 근거가 되며, 설정 범위는 섭씨로 100 (1º)에서 4500 (45º) 사이 입니다.
기본 값은 3800(38º)이며, 설정 범위보다 크거나 작은 값을 입력 시 기본 값 3800 (38º)으로 설정됩니다.
그리고 thermalThresholdLow보다 큰 값을 설정 하여야 합니다.

8. *maskDetectionLevel*

`Visual Face` 기반 마스크 착용 감지 수준을 입력합니다. 여기서 감지 수준은 내부적으로 설정된 값에 근거합니다.

값	설명	기본값
0	착용 감지 안함	기본값
1	감지수준 보통	
2	감지수준 높음	
3	감지수준 매우높음	

9. *auditTemperature*
측정 온도를 이벤트 로그에 기록할지 여부를 설정합니다.

10. *useRejectSound*
thermalThreshold 또는 maskDetectionLevel에 의해서 사용자 인증이 거부되는 경우, 효과음을 발생시킬지 여부를 설정합니다.

11. *useOverlapThermal*
화면상에 열화상 이미지를 중첩해서 표시합니다.

12. *useDynamicROI*
true로 설정하면, 온도 측정 시 고정 영역이 아닌 실제 사용자의 이마를 찾아 측정합니다.

13. *faceCheckOrder*
발열 측정 및 마스크 착용 감지와 인증절차에 대한 순서를 정의합니다.
ID를 조합한 인증이나, PIN을 조합한 인증의 경우, 장치에 접촉하게되는 절차를 거치게 되므로,
고위험군을 고려하는 환경이라면, 발열 측정 후 인증을 선택할지, 그 전에 인증을 수행할지에 대한 선택은 매우 중요한 요소입니다.

값	설명	기본값
0	인증 후 발열 측정 및 마스크 착용 감지	기본값
1	발열 측정 및 마스크 착용 감지 후 인증	
2	인증 절차 생략. 발열 검사 및 마스크 착용 감지만 수행	

BS2ThermalCameraConfig

```
typedef struct {
    uint8_t distance;
    uint8_t emissionRate;

    struct {
        uint16_t x;
        uint16_t y;
        uint16_t width;
        uint16_t height;
    } roi;

    uint8_t useBodyCompensation;
    int8_t compensationTemperature;
} BS2ThermalCameraConfig;
```

1. *distance*
열화상 카메라의 체온 측정 거리를 나타냅니다. 단위는 cm이며 기본값은 100 입니다.

2. *emissionRate*
피사체가 열을 방사하는 비율을 입력합니다.
[95/97/98] 범위내의 입력을 권장합니다. 피사체가 사람인 경우 적정 권장은 98 입니다.

3. *roi*
ROI(Region of interest)는 관심영역을 의미하며, 얼굴에서 발열 측정 시, 관심 대상이되는 영역을 좌표(x, y)와, 범위(width, height) 값을 통해서 지정할 수 있습니다.

4. *useBodyCompensation*
발열 측정된 체온의 보정을 사용할지 여부를 나타냅니다.

5. *compensationTemperature*
실제 체온측정값과, 카메라를 통한 체온측정은 약간의 차이가 존재할 수 있으며, 이곳에 값을 설정하여, 그 차이를 보정할 수 있습니다.
설정할 온도의 10을 곱한 값으로 입력하여야하며, 최소 -50에서 최대 50 사이의 값을 지정할 수 있습니다.

BS2BarcodeConfig

```
typedef struct {
    uint8_t useBarcode;
    uint8_t scanTimeout;
    uint8_t bypassData;
    uint8_t treatAsCSN;

    uint8_t useVisualBarcode;
    uint8_t motionSensitivity;
    uint8_t visualCameraScanTimeout;
    uint8_t reserved[9];
} BS2BarcodeConfig;
```

1. *useBarcode*
`XS2-QR` 만 지원 Barcode 사용여부 flag입니다.

2. *scanTimeout*
Barcode scan 시간을 설정합니다. 단위는 초입니다.
기본값은 4초이며, 4~10초 범위 내로 입력 가능합니다.

값	매크로	설명
4	BS2_BARCODE_TIMEOUT_DEFAULT	기본값
4	BS2_BARCODE_TIMEOUT_MIN	최소값
10	BS2_BARCODE_TIMEOUT_MAX	최대값

3. *bypassData*

[+2.8.2] 읽혀진 barcode 정보를 장치에서 처리하지 않고, 서버로 전송하기 위해 사용합니다.
사용자 인증을 위해 사용자 정보 구조체에 barcode 값을 저장하게되는 경우,
32 byte의 크기 제약(BS2CSNCard data 참조)이 있으나,
BS2_SetBarcodeScanListener를 호출하고, 이 옵션을 사용하면 최대 512 byte크기의 barcode를 서버로 전송할 수 있습니다.

4. *treatAsCSN*
[+2.8.2] Barcode를 일반 CSN 카드와 동일하게 처리할지의 여부를 나타냅니다.
XS2-QR 1.1.3부터 적용되며, false의 경우, 이전과 동일하게 처리됩니다.
이방식은 barcode로 처리될 수 있는 문자셋을 ASCII code 32에서 126까지 자유롭게 지정할 수 있습니다.
(BS2_WriteQRCode의 설명 참조)
true로 설정하면, barcode를 기존 CSN과 동일하게 숫자로만 취급하게 됩니다.
따라서, 특수문자, 영문자가 들어있는 barcode 카드 데이터를 설정하고자 하면 오류를 반환할 수 있습니다.
이 경우, card type만 다르게, CSN 카드와 barcode 데이터를 동일한 값으로 사용하고자하는 사용자의 목적에 부합할 수 있습니다.

5. *useVisualBarcode*
[+ 2.9.1] Visual barcode 사용여부 flag입니다.

지원장치	펌웨어
XS2-Finger	V1.2.0
XS2-Card	V1.2.0
BS3	V1.1.0

Visual barcode는 QR code sensor가 아닌, 일반 visual camera를 통해 QR code 읽기를 지원하며,
이 기능을 사용하기 위해서는, 별도의 라이선스 활성화가 필요합니다.
라이선스 활성화는 BS2_EnableDeviceLicense를 통해 지원합니다.

6. *motionSensitivity*
[+ 2.9.1] Visual barcode에 대한, 모션 센서의 감도를 설정합니다.

값	매크로	설명
0	BS2_MOTION_SENSITIVITY_LOW	낮음
1	BS2_MOTION_SENSITIVITY_NORMAL	중간
2	BS2_MOTION_SENSITIVITY_HIGH	높음

7. *visualCameraScanTimeout*
[+ 2.9.1] Visual camera에대한 scan 시간을 설정합니다. 단위는 초입니다.
기본값은 10초이며, 3~20초 범위 내로 입력 가능합니다.

값	매크로	설명
10	BS2_VISUAL_BARCODE_TIMEOUT_DEFAULT	기본값
3	BS2_VISUAL_BARCODE_TIMEOUT_MIN	최소값
20	BS2_VISUAL_BARCODE_TIMEOUT_MAX	최대값

8. *reserved*
예약된 공간입니다.

BS2InputConfigEx

```
typedef struct {
    uint8_t    numInputs;
    uint8_t    numSupervised;
    uint8_t    reserved[18];

    struct {
        uint8_t    portIndex;
        uint8_t    switchType;
        uint16_t   duration;

        uint8_t    reserved;
        uint8_t    supervisedResistor;
        uint8_t    reserved1[16];

        uint8_t    reserved2[26];
    } inputs[BS2_MAX_INPUT_NUM_EX];

    uint8_t    reserved2[200];
} BS2InputConfigEx;
```

1. *numInputs*
Input 포트의 개수입니다.

2. *numSupervised*
supervised input 포트의 개수입니다.

3. *reserved*
예약된 공간입니다.

4. *portIndex*
Input 포트 번호입니다.

5. *switchType*
Input 신호 유형입니다.

값	설명
0	Normally Open
1	Normally Closed

6. *duration*
Input 신호의 지속시간으로 단위는 밀리초(ms)입니다.

7. *reserved*
예약된 공간입니다.

8. *supervisedResistor*

Supervised input 저항 값의 유형을 설정하거나, 해제할 수(unsupervised) 있습니다.

값	설명
0	1K 저항
1	2.2K 저항
2	4.7K 저항
3	10K 저항
254	Unsupervised(기본값)

9. *reserved1*
예약된 공간입니다.

10. *reserved2*
예약된 공간입니다.

11. *reserved2*
예약된 공간입니다.

BS2RelayActionConfig

```
typedef struct {
    uint32_t        deviceID;               ///< 4 bytes
    uint8_t         reserved[16];            ///< 16 bytes

    struct {
        uint8_t     port;               ///< 1 byte (relay port)
        uint8_t     reserved0;                  ///< 1 byte
        uint8_t     disconnEnabled;             ///< 1 byte (RS485
disconnection)
        uint8_t     reserved[9];             ///< 9 bytes

        struct {
            uint8_t port;               ///< 1 byte (input port)
            uint8_t type;               ///< 1 byte (linkage/latching/release)
            uint8_t mask;               ///< 1 byte (alarm/fault)
            uint8_t reserved[9];             ///< 9 bytes
        } input[BS2_MAX_RELAY_ACTION_INPUT];        ///< 192 bytes
    } relay[BS2_MAX_RELAY_ACTION];              ///< 816 bytes

    uint8_t             reserved2[152];         ///< 152 bytes
} BS2RelayActionConfig;
```

1. *deviceID*
장치 식별번호 입니다.

2. *reserved*
예약된 공간입니다.

3. *relay*

Relay에 대한 설정 정보입니다.

4. *port*
Relay port 번호입니다.

5. *reserved0*
예약된 공간입니다.

6. *disconnEnabled*
true로 설정하면, RS485 연결이 끊겼을때 신호를 발생시키게 됩니다.

7. *reserved*
예약된 공간입니다.

8. *input*
해당 relay port가 어떤 input port들에 대해서 동작할지 정의합니다.

9. *port*
Input port 식별번호 입니다.

10. *type*
해당 input을 어떤 형태의 input으로 동작시킬 것인지 정의합니다.
Linkage로 설정 시 mask로 alarm이 설정되어야 신호가 전달됩니다.

type	값	설명
NONE	0	사용안함
LINKAGE	1	해당 input에 대한 relay로의 연결
LATCHING	2	미지원
RELEASE	3	미지원

11. *mask*
Input 신호정보들에 대해 mask를 설정합니다.

type	값	설명
NONE	0	사용안함
ALARM	1	신호 발생
FAULT	2	단선/단락의 경우 신호 발생

12. *reserved*
예약된 공간입니다.

13. *reserved2*
예약된 공간입니다.

BS2VoipConfigExt

```
typedef struct {
    BS2_USER_ID phoneNumber;
    char description[48 * 3];
    uint8_t reserved[32];
} BS2ExtensionNumber;

typedef struct {
    BS2_BOOL enabled;
    BS2_BOOL useOutboundProxy;
    uint16_t registrationDuration;

    BS2_URL address;
    BS2_PORT port;

    struct {
        uint8_t speaker;      // 0 ~ 100
        uint8_t mic;          // 0 ~ 100
    } volume;             ///< 2 bytes

    BS2_USER_ID id;
    BS2_USER_ID password;
    BS2_USER_ID authorizationCode;

    struct {
        BS2_URL address;
        BS2_PORT port;
        uint8_t reserved[2];
    } outboundProxy;

    uint8_t exitButton;             /// *, #, 0~9
    uint8_t reserved1;
    uint8_t numPhoneBook;
    BS2_BOOL showExtensionNumber;

    BS2ExtensionNumber phonebook[128];

    uint8_t resolution;
    uint8_t transport;
    uint8_t reserved2[30];     ///< 30 bytes (reserved)
} BS2VoipConfigExt;
```

1. *phoneNumber*
내선 번호입니다.

2. *description*
표시 정보입니다.

3. *reserved*
예약된 공간입니다.

4. *enabled*
VoIP extension 기능을 사용할 것인지 여부를 설정합니다.

5. *useOutboundProxy*
Outbound 프록시 서버를 구성할 것인지 여부를 설정합니다.

6. *registrationDuration*
SIP 서버에 관련 정보를 업데이트 시키는 주기입니다.
초단위로 설정하며, 60~600 범위의 값으로 지정하여야 합니다.

7. *address*
SIP 서버(일반적으로 BioStar)의 IP 주소를 입력합니다.

8. *port*
SIP 서버 포트를 입력합니다. 기본 포트는 5060번입니다.

9. *speaker*
인터폰의 스피커 음량 정보를 0에서 100까지 범위로 입력합니다. 기본 값은 50입니다.

10. *mic*
인터폰의 마이크 음량 정보를 0에서 100까지 범위로 입력합니다. 기본 값은 50입니다.

11. *id*
SIP 서버에 연결하는 ID를 입력합니다.
12. *password*
SIP 서버에 연결하는 비밀번호를 지정합니다.

13. *authorizationCode*
SIP 서버에 연결에 필요한 인증코드 값입니다.

14. *outboundProxy*
Outbound 프록시 서버 정보를 입력합니다.

15. *address*
Outbound 프록시 서버의 IP 주소를 입력합니다.

16. *port*
Outbound 프록시 서버 포트를 입력합니다.

17. *reserved*
예약된 공간입니다.

18. *exitButton*
퇴실버튼으로 사용할 버튼 기호입니다.

값	설명
*	'*' ASCII code 42
#	'#' ASCII code 35
0~9	'0'~'9' ASCII code (48~57)

19. *reserved1*
예약된 공간입니다.

20. *numPhoneBook*
전화번호부 갯수입니다.

21. *showExtensionNumber*
전화번호부를 보여줄지 여부를 결정합니다.

22. *phonebook*

128개까지 내선 번호들을 전화번호부로 지정할 수 있습니다.

23. *resolution*
[+2.9.8] 송신 비디오 해상도를 설정합니다.

값	설명
0	360 x 640
1	720 x 480

24. *transport*
[+2.9.8] 데이터 전송 방식을 설정합니다.

값	설명
0	UDP
1	TCP
2	SSL

25. *reserved2*
예약된 공간입니다.

BS2RtspConfig

```
typedef struct {
    BS2_USER_ID id;
    BS2_USER_ID password;

    BS2_URL address;

    BS2_PORT port;
    BS2_BOOL enabled;
    uint8_t reserved;            ///< 1 byte (packing)

    uint8_t resolution;
    uint8_t reserved2[31];        ///< 31 bytes (reserved)
} BS2RtspConfig;
```

1. *id*
RTSP 서버 연결시, 계정 정보입니다.

2. *password*
RTSP 서버 연결시, 비밀번호입니다.

3. *address*
RTSP 서버의 주소를 입력합니다.

4. *port*
RTSP 서버 연결 포트를 입력합니다. 기본 포트는 554입니다.

5. *enabled*
RTSP 연결을 사용할 것인지 여부를 설정합니다.

6. *reserved*
예약된 공간입니다.

7. *resolution*
[+2.9.8] 송신 비디오 해상도를 설정합니다.

값	설명
0	180 x 320
1	720 x 480

8. *reserved2*
예약된 공간입니다.

BS2License

```
typedef struct {
    uint8_t                 index;
    uint8_t                 hasCapability;
    uint8_t                 enable;
    uint8_t                 reserved;
    BS2_LICENSE_TYPE        licenseType;
    BS2_LICENSE_SUB_TYPE    licenseSubType;
    uint32_t                enableTime;
    uint32_t                expiredTime;
    uint32_t                issueNumber;
    uint8_t                 name[BS2_USER_ID_SIZE];
} BS2License;
```

1. *index*
라이선스 인덱스입니다.

2. *hasCapability*
장치가 해당 라이선스를 지원하는지 여부를 나타냅니다.
통상적으로 1을 갖습니다.

3. *enable*
라이선스가 활성화 되어있는지 여부를 나타냅니다.

4. *reserved*
예약된 공간입니다.

5. *licenseType*
라이선스의 종류를 나타냅니다.

값	설명
0x0000	None
0x0001	Visual QR

6. *licenseSubType*
licenseType의 세부 형태를 나타냅니다.

값	설명
0	None
1	Visual QR (CodeCorp)

7. *enableTime*
라이선스 활성화 개시 시간이며, POSIX time으로 표현됩니다.

8. *expiredTime*
라이선스 활성화 종료 시간이며, 0은 제한 없음을 의미합니다.

9. *issueNumber*
발급 고유 번호를 의미합니다.

10. *name*
라이선스 명칭입니다.

BS2LicenseConfig

```
typedef struct {
    uint8_t         version;
    uint8_t         numOfLicense;
    uint8_t         reserved[2];
    BS2License      license[BS2_MAX_LICENSE_COUNT];
    uint8_t         reserved1[16];
} BS2LicenseConfig;
```

1. *version*
라이선스 설정 정보의 버전 정보입니다.

2. *numOfLicense*
등록된 라이선스 갯수를 의미합니다.

3. *reserved*
예약된 공간입니다.

4. *license*
라이선스 정보이며, 최대 16개까지 설정될 수 있습니다.

5. *reserved1*
예약된 공간입니다.

BS2OsdpStandardConfig

```
typedef struct {
    uint32_t                baudRate;             ///< 4 bytes
    uint8_t                    channelIndex;         ///< 1 byte
    uint8_t                    useRegistance;         ///< 1 byte
    uint8_t                    numOfDevices;         ///< 1 byte
    BS2_OSDP_CHANNEL_TYPE     channelType;        ///< 1 byte
    BS2OsdpStandardDevice
slaveDevices[BS2_RS485_MAX_SLAVES_PER_CHANNEL];    ///< 28 * 32 = 896 bytes
    uint8_t                     reserved[4];       ///< 4 bytes
} BS2OsdpStandardChannel;                         ///< 908 bytes

typedef struct {
    uint8_t            mode[BS2_RS485_MAX_CHANNELS_EX];        ///< 8 byte
    uint16_t       numOfChannels;                            ///< 2 byte
    uint8_t        reserved[2];                            ///< 2 bytes
(packing)
    uint8_t        reserved1[32];                            ///< 32 bytes
(reserved)
    BS2OsdpStandardChannel  channels[BS2_RS485_MAX_CHANNELS_EX];     ///<
908 * 8 bytes  = 7264 bytes
} BS2OsdpStandardConfig;                                     ///< 7308 bytes
```

1. *baudRate*
OSDP 장치의 통신 속도이며 설정할 수 있는 범위는 다음과 같습니다.

값
9600
19200
38400
57600
115200

2. *channelIndex*
OSDP 장치가 RS485 통신을 할때 채널 번호입니다.

3. *useRegistance*
종단 저항을 설정할지 결정하는 flag입니다. - 동작에 영향 없음.

4. *numOfDevices*
슬레이브 장치의 개수입니다.

5. *channelType*
RS485 통신을 하는 장치가 어떠한 방식으로 연결되어 있는지를 나타냅니다.
CoreStation40 기준으로, 할당 가능한 채널은 0~4까지 5개인데, 각 채널내에서는 슈프리마 장치와 OSDP 장치가 혼용되어 운용될어질 수 없습니다.
특정 채널에 연결된 장치가 없는 경우, Suprema 장치든, OSDP 장치이든 연결될 수 있음을 나타내는 0을 갖습니다.
만일 특정 채널로 Suprema 장치가 연결되면, 이후 해당 채널은 Suprema 장치들의 연결만 허용되며, channelType은 1을 갖습니다. OSDP 장치가 연결되어 있더라도 무시되어집니다.
만일 특정 채널로 OSDP 장치가 연결되면, 이후 해당 채널은 OSDP 장치 연결만 허용되며, channelType 은 2를 갖습니다. Suprema 장치가 연결되어 있더라도 무시되어집니다.
CoreStation40 각각의 채널들은 Suprema 장치 채널, OSDP 장치 채널들로 혼용되어 운용될 수 있습니다.
채널에 OSDP로 장치가 연결이되면 허용이 가능한 최대 장치수는 2대로 제한되며, 이미 해당 채널이 최대치에 도달했으면 channelType은 3을 갖으며 더이상 연결이 불가함을 나타냅니다.

값	설명
0	Normal
1	Suprema 장치
2	OSDP 장치
3	OSDP 장치 FULL

6. *slaveDevices*
채널내의 슬레이브 장치 정보입니다.

7. *reserved*
예약된 공간입니다.

8. *mode*
RS485 네트워크에서 어떤 모드로 동작할지 결정하는 flag인데, 문서 작성일 2023/1/12일 기준으로 Osdp standard config를 지원하는 장치는 CoreStation40 뿐이므로 항상 master 값을 갖습니다.

값	설명
0	사용안함
1	Master
2	Slave
3	Standalone (기본값)

9. *numOfChannels*
채널의 개수입니다. CoreStation40은 총 5개의 채널을 갖습니다.

10. *reserved*
예약된 공간입니다.

11. *reserved1*
예약된 공간입니다.

12. *channels*
채널별 OSDP 장치정보입니다.
최대 8개까지 채널정보를 가질 수 있으나, CoreStation40이 5개의 채널을 가지므로 0~4번까지만 유의미합니다.

BS2OsdpStandardActionConfig

```
typedef struct{
    BS2_BOOL                            use;            ///< 1 byte
    uint8_t                             readerNumber;   ///< 1 byte
    uint8_t                             ledNumber;      ///< 1 byte

    BS2_OSDP_STANDARD_LED_COMMAND       tempCommand;    ///< 1 byte
    uint8_t                             tempOnTime;     ///< 1 byte
    uint8_t                             tempOffTime;    ///< 1 byte
    BS2_OSDP_STANDARD_COLOR             tempOnColor;    ///< 1 byte
    BS2_OSDP_STANDARD_COLOR             tempOffColor;   ///< 1 byte
    uint16_t                            tempRunTime;    ///< 2 bytes
    BS2_OSDP_STANDARD_LED_COMMAND      permCommand;    ///< 1 byte
    uint8_t                            permOnTime;     ///< 1 byte
    uint8_t                            permOffTime;    ///< 1 byte
    BS2_OSDP_STANDARD_COLOR            permOnColor;    ///< 1 byte
    BS2_OSDP_STANDARD_COLOR            permOffColor;   ///< 1 byte

    uint8_t                            reserved;       ///< 1 byte
} BS2OsdpStandardLedAction;            ///< 16 bytes

typedef struct {
    BS2_BOOL                   use;            ///< 1 byte
    uint8_t                    readerNumber;   ///< 1 byte
    BS2_OSDP_STANDARD_TONE     tone;           ///< 1 byte
    uint8_t                    onTime;         ///< 1 byte
    uint8_t                    offTime;        ///< 1 byte
    uint8_t                    numOfCycle;     ///< 1 byte
    uint8_t                    reserved[2];    ///< 2 bytes
} BS2OsdpStandardBuzzerAction;                 ///< 8 bytes

typedef struct {
    BS2_OSDP_STANDARD_ACTION_TYPE   actionType;     ///< 1 byte
    uint8_t                         reserved[3];    ///< 3 bytes
    BS2OsdpStandardLedAction        led[2];         ///< 16 x 2 = 32 bytes
    BS2OsdpStandardBuzzerAction     buzzer;         ///< 8 bytes
} BS2OsdpStandardAction;                            ///< 44 bytes

typedef struct
{
    uint8_t                 version;                    ///< 1 byte
    uint8_t                 reserved[3];                ///< 3 byes
    BS2OsdpStandardAction   actions[BS2_OSDP_STANDARD_ACTION_MAX_COUNT];
///< 44 x 32  = 1408
} BS2OsdpStandardActionConfig;                          ///< 1412 bytes
```

1. *use*
해당 LED action을 사용할 것인지 여부를 나타냅니다.

2. *readerNumber*
OSDP 장치의 순번입니다.

3. *ledNumber*
OSDP 장치가 갖고 있는 LED 순번입니다.

4. *tempCommand*
Temporary command 입니다.

값	설명
0	No Operation
1	Cancel
2	Set

5. *tempOnTime*
Temporary command에 대해서 LED가 켜져있는 시간을 의미하며, 100ms 단위로 설정합니다.
예를 들어 2초간 LED on 상태를 유지시키려면 20을 입력하면 됩니다.

6. *tempOffTime*
Temporary command에 대해서 LED가 꺼져있는 시간을 의미하며, 100ms 단위로 설정합니다.
예를 들어 1초간 LED off 상태를 유지시키려면 10을 입력하면 됩니다.

7. *tempOnColor*
Temporary command에 대해서 On 상태의 LED 색상을 지정합니다.

값	설명
0	BLACK
1	RED
2	GREEN
3	AMBER
4	BLUE
5	MAGENTA
6	CYAN
7	WHITE

8. *tempOffColor*
Temporary command에 대해서 Off 상태의 LED 색상을 지정합니다.

값	설명
0	BLACK
1	RED
2	GREEN
3	AMBER
4	BLUE
5	MAGENTA
6	CYAN
7	WHITE

9. *tempRunTime*
Temporary command에 대해 LED On/Off를 얼마나 지속시킬지를 100ms 단위로 설정합니다.
tempOnTime/tempOffTime, tempOnColor/tempOffColor로 설정한 색상과 시간으로 번갈아 깜빡이며, 이 깜빡임은 tempRunTime 만큼 유지됩니다.

10. *permCommand*
Permanent command 입니다. 11. *permOnTime*
Permanent command에 대해서 LED가 켜져있는 시간을 의미하며, 100ms 단위로 설정합니다.

12. *permOffTime*
Permanent command에 대해서 LED가 꺼져있는 시간을 의미하며, 100ms 단위로 설정합니다.

13. *permOnColor*
Permanent command에 대해서 On 상태의 LED 색상을 지정합니다.

14. *permOffColor*

Permanent command에 대해서 Off 상태의 LED 색상을 지정합니다.

15. *reserved*
예약된 공간입니다.

16. *use*
해당 tone action을 사용할 것인지 여부를 나타냅니다.

17. *readerNumber* OSDP 장치의 순번입니다.

18. *tone*
Buzzer를 설정합니다.

값	설명
0	None
1	Off
2	On

19. *onTime*
tone에 대해서 On 상태 유지시간을 100ms 단위로 설정합니다.

20. *offTime*
tone에 대해서 Off 상태 유지시간을 100ms 단위로 설정합니다.

21. *numOfCycle*
tone On/Off를 얼마나 반복할지 횟수를 지정합니다. 0으로 설정 시 무한 반복을 의미합니다.

22. *reserved*
예약된 공간입니다.

23. *actionType*
action을 지정합니다.

값	설명
0	None
1	Success
2	Fail
3	Wait input

24. *reserved*
예약된 공간입니다.

25. *led*
OSDP 장치의 LED 설정 정보입니다.

26. *buzzer*
OSDP 장치의 buzzer 설정 정보입니다.

27. *version*
Action configuration에 대한 버전 정보입니다. 현재는 0입니다.

28. *reserved*
예약된 공간입니다.

29. *actions*
OSDP 장치에 대한 LED/buzzer 정보이며, 최대 32개 까지 지정할 수 있습니다.

BS2CustomMifareCard

```
typedef struct {
    uint8_t primaryKey[6];
    uint8_t reserved1[2];
    uint8_t secondaryKey[6];
    uint8_t reserved2[2];
    uint16_t startBlockIndex;
    uint8_t dataSize;
    uint8_t skipBytes;
    uint8_t reserved[4];
} BS2CustomMifareCard;
```

1. *primaryKey*
Mifare card 정보에 접근하기 위한 첫번째 암호화 키입니다.

2. *reserved1*
예약된 공간입니다.

3. *secondaryKey*
Mifare card 정보에 접근하기 위한 두번째 암호화 키입니다.

4. *reserved2*
예약된 공간입니다.

5. *startBlockIndex*
Mifare data storage에서의 start block index입니다.

6. *dataSize*
카드 데이터가 몇 byte의 크기를 갖는지를 나타냅니다.

7. *skipBytes*
카드 데이터가 나타나는 위치를 의미합니다.
카드 데이터를 읽어들일 시작점 입니다. 처음부터 읽을경우 0이고, 첫번째 이후일 경우 건너뛴 byte 수를 나타냅니다.

8. *reserved*
예약된 공간입니다.

BS2CustomDesFireCard

```
typedef struct {
    uint8_t primaryKey[16];
    uint8_t secondaryKey[16];
    uint8_t appID[3];
    uint8_t fileID;
    uint8_t encryptionType;                    // 0: DES/3DES, 1: AES
    uint8_t operationMode;                     // 0: legacy(use picc master
key), 1: new mode(use app master, file read, file write key)
    uint8_t dataSize;
    uint8_t skipBytes;
    uint8_t reserved[4];
    BS2DesFireAppLevelKey desfireAppKey;        ///<52 bytes
} BS2CustomDesFireCard;                          ///<96 Bytes
```

1. *primaryKey*
DesFire card 정보에 접근하기 위한 첫번째 암호화 키입니다. (일반설정)

2. *secondaryKey*
DesFire card 정보에 접근하기 위한 두번째 암호화 키입니다. (일반설정)

3. *appID*
사용자 인증을 위해 DESFire 카드내에 내장된 사용할 애플리케이션 식별자입니다.

4. *fileID*
DESFire 카드내에 내장된 사용할 애플리케이션이 데이터를 읽고 쓰기 위한 파일 식별자입니다.

5. *encryptionType*
데이터 암호화 방식을 나타냅니다.

값	설명
0	DES/3DES
1	AES

6. *operationMode*
운영 방식을 지정합니다.

값	설명
0	일반설정 사용 (PICC master key를 사용)
1	고급설정 사용 (App master key를 사용)

7. *dataSize*
카드 데이터가 몇 byte의 크기를 갖는지를 나타냅니다.

8. *skipBytes*
카드 데이터가 나타나는 위치를 의미합니다.
카드 데이터를 읽어들일 시작점 입니다. 처음부터 읽을경우 0이고, 첫번째 이후일 경우 건너뛴 byte 수를 나타냅니다.

9. *reserved*
예약된 공간입니다.

10. *desfireAppKey*
DesFire card 정보에 접근하기 위한 키 정보를 나타냅니다. (고급설정)

BS2CustomCardConfig

```
typedef struct {
    uint8_t primaryKey[16];
    uint8_t secondaryKey[16];
    uint16_t startBlockIndex;
    uint8_t dataSize;
    uint8_t skipBytes;
    uint8_t reserved[16];
} BS2CustomMifareCardEx;

typedef struct {
    BS2_CARD_DATA_TYPE dataType;
    BS2_BOOL useSecondaryKey;
    uint8_t reserved1[2];

    BS2CustomMifareCard mifare;
    BS2CustomDesFireCard desfire;
    BS2CustomMifareCardEx mifareEx;
    BS2_MIFARE_ENCRYPTION mifareEncType;
    uint8_t reserved3[67];

    BS2_CARD_BYTE_ORDER smartCardByteOrder;
    uint8_t reserved4[3];
    BS2_UID formatID;
    uint8_t reserved5[8];
} BS2CustomCardConfig;
```

1. *primaryKey*
Mifare card 정보에 접근하기 위한 첫번째 암호화 키입니다.

2. *secondaryKey*
Mifare card 정보에 접근하기 위한 두번째 암호화 키입니다.

3. *startBlockIndex*
Mifare data storage에서의 start block index입니다.

4. *datasize*
카드 데이터가 몇 byte의 크기를 갖는지를 나타냅니다.

5. *skipBytes*
카드 데이터가 나타나는 위치를 의미합니다.
카드 데이터를 읽어들일 시작점 입니다. 처음부터 읽을경우 0이고, 첫번째 이후일 경우 건너뛴 byte 수를 나타냅니다.

6. *reserved*
예약된 공간입니다.

7. *dataType*
Card의 데이터 유형입니다.

값	설명
0	바이너리
1	아스키
2	UTF16
3	BCD

8. *useSecondaryKey*
두번째 암호화 키 사용 유무를 결정하는 flag입니다.

9. *reserved1*
예약된 공간입니다.

10. *mifare*
Mifare custom card 정보를 설정합니다.

11. *desfire*
DESFire custom card 정보를 설정합니다.

12. *mifareEx*
[+2.9.9] 장치의 Mifare Custom Card 중에서 AES128 암호화 된 카드에 사용되는 설정 정보를 저장합니다.

13. *mifareEncType*
[+2.9.9] Mifare Custom Card의 암호화 방식을 지정합니다. Mifare Classic 카드는 CRYPTO1만 지원하며, Mifare Plus 카드는 CRYPTO1과 AES128 모두 지원합니다.
CRYPTO1으로 설정하면 BS2CustomMifareCard의 Mifare 설정이 적용되고, AES128로 설정하면 BS2CustomMifareConfigEx의 mifareEx 설정이 적용됩니다.

값	설명
0	CRYPTO1
1	AES128

14. *reserved3*
예약된 공간입니다.

15. *smartCardByteOrder*
MSB 또는 LSB로 출력을 지정할 수 있습니다.

값	설명
0	MSB
1	LSB

16. *reserved4*
예약된 공간입니다.

17. *formatID*
BioStar 2 애플리케이션에서 card configuration을 데이터베이스 관리할 필요가 있을 경우 사용할 수 있는 식별자입니다.

18. *reserved5*
예약된 공간입니다.

BS2MifareCardConfigEx

```
typedef struct {
    uint8_t primaryKey[16];
    uint8_t secondaryKey[16];
    uint16_t startBlockIndex;
    uint8_t reserved[14];
} BS2MifareCardEx;

typedef struct {
    BS2MifareCardEx mifareEx;
    uint8_t reserved[16];
} BS2MifareCardConfigEx;
```

1. *primaryKey*
Mifare card 정보에 접근하기 위한 첫번째 암호화 키입니다.

2. *secondaryKey*
Mifare card 정보에 접근하기 위한 두번째 암호화 키입니다.

3. *startBlockIndex*
Mifare data storage에서의 start block index입니다.

4. *reserved*
예약된 공간입니다.

1) 최대 전송 단위, Maximum Transmission Unit
2) 최상위 비트, Most Significant Bit
3) 최하위 비트, Least Significant Bit