

File created: 11-Jan-89 14:52:06 {NB:PARC:XEROX}<NFS>SOURCES>RPCDECLS.;2

changes to: (IL:VARS IL:RPCDECLSCOMS)
(IL:VARIABLES *RPC-ACCEPT-PROGRAM-UNAVAILABLE* *PORTMAPPER-SOCKET*)

previous date: 19-Oct-88 18:29:59 {NB:PARC:XEROX}<NFS>SOURCES>RPCDECLS.;1

Read Table: XCL

Package: RPC2

Format: XCCS

; Copyright (c) 1988, 1989 by Xerox Corporation. All rights reserved.

(IL:RPAQQ **IL:RPCDECLSCOMS**

;; Macros useful for low-level RPC hacking.

```
(IL:PROPS (IL:RPCDECLS IL:MAKEFILE-ENVIRONMENT IL:FILETYPE))
(IL:FUNCTIONS GETBASE-INTEGERS GETBASE-UNSIGNED INTEGER-FROM-BYTES
  UNSIGNED-FROM-SIGNED PUTBASE-INTEGERS FOLDLO UNFOLD VECTOR-BASE VECTOR-OFFSET PADDING-BYTES)
  ; Call methods
(IL:FUNCTIONS RPC-METHOD RPC-CALL-METHOD REINITIALIZE-RPCSTREAM GETBYTE GETRAWBYTES SKIPBYTES GETCELL
  GETOFFSET PUTBYTE PUTRAWBYTES ZEROBYTES PUTCELL GETUNSIGNED PUTUNSIGNED)
(IL:VARIABLES *WORDS-PER-CELL* *BYTES-PER-CELL* *BYTES-PER-WORD*)
  ; Well-known RPC constants
(IL:VARIABLES *RPC-MSG-CALL* *RPC-MSG-REPLY* *RPC-REPLY-ACCEPTED* *RPC-REPLY-REJECTED*
  *RPC-ACCEPT-SUCCESS* *RPC-ACCEPT-PROGRAM-UNAVAILABLE* *RPC-VERSION*
  *INTERNAL-TIME-UNITS-PER-MSEC* *PORTMAPPER-SOCKET*)
  ; For those that need IP/TCP stuff
(IL:FUNCTIONS LOAD-TCP-EXPORTS))
```

;; Macros useful for low-level RPC hacking.

(IL:PUTPROPS **IL:RPCDECLS IL:MAKEFILE-ENVIRONMENT** (:READTABLE "XCL" :PACKAGE "RPC2"))

(IL:PUTPROPS **IL:RPCDECLS IL:FILETYPE** :COMPILE-FILE)

(DEFMACRO **GETBASE-INTEGERS** (BASE BYTEOFFSET)

```
"Interpret 32 bits at BYTEOFFSET from BASE as a signed integer."
` (LET ((BASE (IL:\ADDBASE ,BASE (FOLDLO ,BYTEOFFSET *BYTES-PER-WORD*)))
  (IL:\MAKENUMBER (IL:\GETBASE BASE 0)
  (IL:\GETBASE BASE 1))))
```

(DEFMACRO **GETBASE-UNSIGNED** (BASE BYTEOFFSET)

```
"Interpret 32 bits at BYTEOFFSET from BASE as an unsigned integer."
;; This differs from GETBASE-INTEGERS only when the high bit is on, in which case we are forced to make (choke) a bignum, which we try to do
;; efficiently.
```

```
` (LET* ((BASE (IL:\ADDBASE ,BASE (FOLDLO ,BYTEOFFSET *BYTES-PER-WORD*)))
  (HI (IL:\GETBASE BASE 0)))
  (IF (> HI 32767)
    (BIGNUM-MAKE-NUMBER HI (IL:\GETBASE BASE 1))
    (IL:\MAKENUMBER HI (IL:\GETBASE BASE 1))))
```

(DEFMACRO **INTEGER-FROM-BYTES** (BYTE0 BYTE1 BYTE2 BYTE3)

```
"Interprets these 32 bits as a signed integer"
` (IL:\MAKENUMBER (+ (UNFOLD ,BYTE0 256)
  ,BYTE1)
  (+ (UNFOLD ,BYTE2 256)
  ,BYTE3))
```

(DEFMACRO **UNSIGNED-FROM-BYTES** (BYTE0 BYTE1 BYTE2 BYTE3)

```
"Interprets these 32 bits as an unsigned integer"
` (LET* ((HI (+ (UNFOLD ,BYTE0 256)
  ,BYTE1))
  (LO (+ (UNFOLD ,BYTE2 256)
  ,BYTE3)))
  (IF (> HI 32767)
    (BIGNUM-MAKE-NUMBER HI LO)
    (IL:\MAKENUMBER HI LO))))
```

(DEFMACRO **UNSIGNED-FROM-SIGNED** (VALUE)

```
"Interpret the 32 bits of VALUE's representation as an unsigned integer."
` (LET ((VALUE ,VALUE))
  (IF (> 0 VALUE)
    (+ VALUE TWOTO32ND)
    VALUE))
```

(DEFMACRO **PUTBASE-INTEGERS** (BASE BYTEOFFSET VALUE)

```
"Store integer VALUE at BYTEOFFSET bytes beyond BASE."
```

;; Note this handles both "signed" and "unsigned" numbers. We do type analysis here to avoid gratuitous consing when handling anything large.

```
`(LET ((BASE (IL:\\ADDBASE ,BASE (FOLDLO ,BYTEOFFSET *BYTES-PER-WORD*)))
      (VALUE ,VALUE))
  (COND
   ((IL:SMALLP VALUE) ; An immediate value
    (IL:\\PUTBASE BASE 0 (IF (< VALUE 0)
                          65535
                          0))
    (IL:\\PUTBASE BASE 1 (IL:\\LOLOC VALUE)))
   ((EQ (IL:NTYPX VALUE)
        IL:\\FIXP) ; A 32-bit integer box--just blt it
    (IL:\\BLT BASE VALUE 2))
   (T (PUTBASE-BIGNUM BASE VALUE))))
```

```
(DEFMACRO FOLDLO (FORM DIVISOR)
  (LET ((DIV (IF (CONSTANTP DIVISOR)
                (EVAL DIVISOR)
                DIVISOR)))
    (OR (AND DIV (IL:POWEROFTWOP DIV))
        (IL:\\ILLEGAL.ARG DIV))
    (LIST 'IL:LRSH FORM (IL:SUB1 (IL:INTEGERLENGTH DIV))))))
```

```
(DEFMACRO UNFOLD (FORM DIVISOR)
  (LET ((DIV (IF (CONSTANTP DIVISOR)
                (EVAL DIVISOR)
                DIVISOR)))
    (OR (AND DIV (IL:POWEROFTWOP DIV))
        (IL:\\ILLEGAL.ARG DIV))
    (LIST 'IL:LLSH FORM (IL:SUB1 (IL:INTEGERLENGTH DIV))))))
```

```
(DEFMACRO VECTOR-BASE (VECTOR)
  "Get raw string/vector base address. Use VECTOR-OFFSET, too, unless you know this is a brand new one without displacement."
  `(IL:|fetch| (IL:ONED-ARRAY IL:BASE) IL:|of| ,VECTOR))
```

```
(DEFMACRO VECTOR-OFFSET (VECTOR)
  "Get raw vector offset. Interpretation depends on element type, of course."
  `(IL:|fetch| (IL:ONED-ARRAY IL:OFFSET) IL:|of| ,VECTOR))
```

```
(DEFMACRO PADDING-BYTES (BYTECOUNT)
  "Returns number of bytes needed to pad BYTECOUNT bytes out to a multiple of 32 bits."
  `(LET ((N ,BYTECOUNT))
    (- (LOGAND (+ N 3)
               -4)
       N)))
```

;; Call methods

```
(DEFMACRO RPC-METHOD (OP STREAM)
  "Returns the function that implements OP (unevaluated) on STREAM."
  `(,(INTERN (CONCATENATE 'STRING "RPC-METHODS-" (STRING OP))
            "RPC2")
    (RPC-STREAM-METHODS ,STREAM)))
```

```
(DEFMACRO RPC-CALL-METHOD (OP &REST ARGS)
  "Invoke the OP method on ARGS, the first of which must be the RPC-STREAM that defines the method"
  `(FUNCALL (RPC-METHOD ,OP ,(FIRST ARGS))
            ,@ARGS))
```

```
(DEFMACRO REINITIALIZE-RPCSTREAM (STREAM DESTADDR DESTSOCKET)
  "Reuse an existing RPC Stream to send a new packet. Resets length counters, reinitializes packets, etc."
  `(RPC-CALL-METHOD INITIALIZE ,STREAM ,DESTADDR ,DESTSOCKET))
```

```
(DEFMACRO GETBYTE (XDRSTREAM)
  "Applies the GETBYTE method of an RPC Stream to read in and return the next byte of the stream."
  `(RPC-CALL-METHOD GETBYTE ,XDRSTREAM))
```

```
(DEFMACRO GETRAWBYTES (XDRSTREAM BASE OFFSET NBYTES)
  "Applies the GETRAWBYTES method of an RPC stream to read NBYTES bytes from the stream to BASE,OFFSET."
  `(RPC-CALL-METHOD GETRAWBYTES ,XDRSTREAM ,BASE ,OFFSET ,NBYTES))
```

```
(DEFMACRO SKIPBYTES (RPCSTREAM NBYTES)
  "Applies the SKIPBYTES method of an RPC stream to skip NBYTES bytes of input."
  `(RPC-CALL-METHOD SKIPBYTES ,RPCSTREAM ,NBYTES))
```

```

(DEFMACRO GETCELL (XDRSTREAM)
  "Applies the GETCELL method of an RPC Stream to read in and return the next cell of the stream. A cell is a
  32-bit two's complement integer."
  `(RPC-CALL-METHOD GETCELL ,XDRSTREAM))

(DEFMACRO GETOFFSET (XDRSTREAM)
  "Returns dotted pair (base . byteoffset), pointing at current position in incoming packet"
  `(RPC-CALL-METHOD GETOFFSET ,XDRSTREAM))

(DEFMACRO PUTBYTE (RPCSTREAM VALUE)
  "Applies the PUTBYTE method of an RPC Stream to write the byte VALUE on that stream. VALUE is an integer
  between 0 and 255 inclusive."
  `(RPC-CALL-METHOD PUTBYTE ,RPCSTREAM ,VALUE))

(DEFMACRO PUTRAWBYTES (RPCSTREAM BASE OFFSET NBYTES)
  "Applies the PUTRAWBYTES method of an RPC stream to write the NBYTES bytes from BASE,OFFSET to the stream."
  `(RPC-CALL-METHOD PUTRAWBYTES ,RPCSTREAM ,BASE ,OFFSET ,NBYTES))

(DEFMACRO ZEROBYTES (RPCSTREAM NBYTES)
  "Applies the ZEROBYTES method of an RPC stream to write NBYTES bytes of zero to the output."
  `(RPC-CALL-METHOD ZEROBYTES ,RPCSTREAM ,NBYTES))

(DEFMACRO PUTCELL (RPCSTREAM VALUE)
  "Applies the PUTCELL method of an RPC Stream to write the cell VALUE on that stream. A cell is a 32-bit two's
  complement integer."
  `(RPC-CALL-METHOD PUTCELL ,RPCSTREAM ,VALUE))

(DEFMACRO GETUNSIGNED (RPCSTREAM)
  "Fetch an unsigned 32-bit integer from RPCSTREAM. Uses the GETUNSIGNED method."
  `(RPC-CALL-METHOD GETUNSIGNED ,RPCSTREAM))

(DEFMACRO PUTUNSIGNED (RPCSTREAM VALUE)
  "Write a 32-bit unsigned integer to RPCSTREAM. Uses PUTCELL method."
  ;; Note that no coercion is need here, because the bits of the bignum are the same as the bits of the signed integer.
  `(PUTCELL ,RPCSTREAM ,VALUE))

(DEFCONSTANT *WORDS-PER-CELL* 2
  "The number of words (16 bits) per cell.")

(DEFCONSTANT *BYTES-PER-CELL* 4
  "Number of 8-bit bytes per RPC cell.")

(DEFCONSTANT *BYTES-PER-WORD* 2)

;; Well-known RPC constants

(DEFCONSTANT *RPC-MSG-CALL* 0
  "Constant 0 in packet means RPC call, 1 means reply")

(DEFCONSTANT *RPC-MSG-REPLY* 1)

(DEFCONSTANT *RPC-REPLY-ACCEPTED* 0
  "Switch in reply body")

(DEFCONSTANT *RPC-REPLY-REJECTED* 1
  "Switch in reply body")

(DEFCONSTANT *RPC-ACCEPT-SUCCESS* 0
  "Switch in accepted reply.")

(DEFCONSTANT *RPC-ACCEPT-PROGRAM-UNAVAILABLE* 1)

(DEFCONSTANT *RPC-VERSION* 2
  "This code will only work for SUN RPC Version 2")

(DEFCONSTANT *INTERNAL-TIME-UNITS-PER-MSEC* (/ INTERNAL-TIME-UNITS-PER-SECOND 1000)
  "This gets used in EXCHANGE-UDP-PACKETS.")

```

```
(DEFCONSTANT *PORTMAPPER-SOCKET* 111
  "Well-known socket for portmapper")
```

```
:: For those that need IP/TCP stuff
```

```
(DEFUN LOAD-TCP-EXPORTS ()
  (PROG1 (IL:FILESLOAD (IL:SOURCE)
              IL:TCPEXPORTS)
    ; Now stop us from loading it again (This really ought to be on
    ; TCPEXPORTS)
    (OR (GET 'IL:TCPEXPORTS 'IL:FILEDATES)
        (SETF (GET 'IL:TCPEXPORTS 'IL:FILEDATES)
              T))))
(IL:PUTPROPS IL:RPCDECLS IL:COPYRIGHT ("Xerox Corporation" 1988 1989))
```

FUNCTION INDEX

LOAD-TCP-EXPORTS4

MACRO INDEX

FOLDLO2	GETUNSIGNED3	PUTUNSIGNED3	UNSIGNED-FROM-SIGNED1
GETBASE-INTEGER1	INTEGER-FROM-BYTES1	REINITIALIZE-RPCSTREAM ..2	VECTOR-BASE2
GETBASE-UNSIGNED1	PADDING-BYTES2	RPC-CALL-METHOD2	VECTOR-OFFSET2
GETBYTE2	PUTBASE-INTEGER1	RPC-METHOD2	ZEROBYTES3
GETCELL3	PUTBYTE3	SKIPBYTES2	
GETOFFSET3	PUTCELL3	UNFOLD2	
GETRAWBYTES2	PUTRAWBYTES3	UNSIGNED-FROM-BYTES1	

CONSTANT INDEX

BYTES-PER-CELL3	*RPC-ACCEPT-PROGRAM-UNAVAILABLE* .3	*RPC-REPLY-ACCEPTED*3
BYTES-PER-WORD3	*RPC-ACCEPT-SUCCESS*3	*RPC-REPLY-REJECTED*3
INTERNAL-TIME-UNITS-PER-MSEC ...3	*RPC-MSG-CALL*3	*RPC-VERSION*3
PORTMAPPER-SOCKET4	*RPC-MSG-REPLY*3	*WORDS-PER-CELL*3

PROPERTY INDEX

IL:RPCDECLS1
