OVERVIEW

This is a comparison of the instruction set from the ATMEL Atmega328 and the LGT8Fx. My work can be seen on the page titled "CompareWorkbook" inside the "AVRvsLGTInstructionset" spreadsheet. On that page I carefully extracted the tables directly from all three PDF datasheets and compared them line by line. I used the ATMEL list order and rearranged the LGT instructions to compare them 1:1. The results of this comparison are shown below. In short, the LGT has 3 missing instructions and 4 new instructions not present in the Atmega328. Several instructions also claim to take fewer cycles with the LGT.

Instructions present in the Atmega328 but NOT in the LGT8F328 :

Mnemonics	*Operands*	*Description*	*Operation*	*Flags*	*#Clocks*
SEH		Set Half Carry Flag in SREG	H ← 1	H	1
CLH		Clear Half Carry Flag in SREG	H ← 0	H	1
SPM		Store Program Memory	(Z) ← R1:R0	None	-

NEW instructions only present in the LGT8F328 :

Mnemonics	*Operands*	*Description*	*Operation*	*Flags*	*#Clocks*
LD	Rd, Z+	间接加载,地址递增 || Indirect loading, address increment	Rd ← (Z), Z ← Z+1	None	1
LD	Rd, -Z	地址递减,间接加载 || Decrement of address, indirect loading	Z ← Z - 1, Rd ← (Z)	None	1
LDD	Rd, Z+q	带偏移量的间接加载 || Indirect loading with offset	Rd ← (Z + q)	None	1
LDS	Rd, k	直接从SRAM中加载 || Load directly from SRAM	Rd ← (k)	None	2

Instructions present in the Atmega328 AND in the LGT8F328, but claiming to take less clock cycles.

Mnemonics	*Operands*	*Description*	*Operation*	*Flags*	*#CLK ATMEL*	*#CLK LGT*
ADIW	Rdl,K	Add Immediate to Word	Rdh:Rdl ← Rdh:Rdl + K	Z,C,N,V,S	2	1
SBIW	Rdl,K	Subtract Immediate from Word	Rdh:Rdl ← Rdh:Rdl - K	Z,C,N,V,S	2	1
MUL	Rd,Rr	Multiply Unsigned	R1:R0 ← Rd x Rr	Z,C	2	1
MULS	Rd,Rr	Multiply Signed	R1:R0 ← Rd x Rr	Z,C	2	1
MULSU	Rd,Rr	Multiply Signed with Unsigned	R1:R0 ← Rd x Rr	Z,C	2	1
FMUL	Rd,Rr	Fractional Multiply Unsigned	R1:R0 ← (Rd x Rr) << 1	Z,C	2	1
FMULS	Rd,Rr	Fractional Multiply Signed	R1:R0 ← (Rd x Rr) << 1	Z,C	2	1
FMULSU	Rd,Rr	Fractional Multiply Signed with Unsigned	R1:R0 ← (Rd x Rr) << 1	Z,C	2	1
RJMP	k	Relative Jump	PC ← PC + k + 1	None	2	1
JMP(1)	k	Direct Jump	PC ← k	None	3	2
RCALL	k	Relative Subroutine Call	PC ← PC + k + 1	None	3	1
ICALL		Indirect Call to (Z)	PC ← Z	None	3	2
CALL(1)	k	Direct Subroutine Call	PC ← k	None	4	2
RET		Subroutine Return	PC ← STACK	None	4	2
RETI		Interrupt Return	PC ← STACK	I	4	2
CPSE	Rd,Rr	Compare, Skip if Equal	if (Rd = Rr) PC ← PC + 2 or 3	None	1/2/3	1/2
SBRC	Rr,b	Skip if Bit in Register Cleared	if (Rr(b)=0) PC ← PC + 2 or 3	None	1/2/3	1/2
SBRS	Rr,b	Skip if Bit in Register is Set	if (Rr(b)=1) PC ← PC + 2 or 3	None	1/2/3	1/2
SBIC	P,b	Skip if Bit in I/O Register Cleared	if (P(b)=0) PC ← PC + 2 or 3	None	1/2/3	1/2
SBIS	P,b	Skip if Bit in I/O Register is Set	if (P(b)=1) PC ← PC + 2 or 3	None	1/2/3	1/2
SBI	P,b	Set Bit in I/O Register	I/O(P,b) ← 1	None	2	1
CBI	P,b	Clear Bit in I/O Register	I/O(P,b) ← 0	None	2	1
LD	Rd,X	Load Indirect	Rd ← (X)	None	2	1/2
LD	Rd,X+	Load Indirect and Post-Inc.	Rd ← (X), X ← X + 1	None	2	1/2
LD	Rd,-X	Load Indirect and Pre-Dec.	X ← X - 1, Rd ← (X)	None	2	1/2
LD	Rd,Y	Load Indirect	Rd ← (Y)	None	2	1/2
LD	Rd,Y+	Load Indirect and Post-Inc.	Rd ← (Y), Y ← Y + 1	None	2	1/2
LD	Rd,-Y	Load Indirect and Pre-Dec.	Y ← Y - 1, Rd ← (Y)	None	2	1/2
LDD	Rd,Y+q	Load Indirect with Displacement	Rd ← (Y + q)	None	2	1/2
LD	Rd,Z	Load Indirect	Rd ← (Z)	None	2	1/2
LD	Rd,Z+	Load Indirect and Post-Inc.	Rd ← (Z), Z ← Z+1	None	2	1/2
LD	Rd,-Z	Load Indirect and Pre-Dec.	Z ← Z - 1, Rd ← (Z)	None	2	1/2
LDD	Rd,Z+q	Load Indirect with Displacement	Rd ← (Z + q)	None	2	1/2
ST	X,Rr	Store Indirect	(X) ← Rr	None	2	1
ST	X+,Rr	Store Indirect and Post-Inc.	(X) ← Rr, X ← X + 1	None	2	1
ST	-X,Rr	Store Indirect and Pre-Dec.	X ← X - 1, (X) ← Rr	None	2	1
ST	Y,Rr	Store Indirect	(Y) ← Rr	None	2	1
ST	Y+,Rr	Store Indirect and Post-Inc.	(Y) ← Rr, Y ← Y + 1	None	2	1
ST	-Y,Rr	Store Indirect and Pre-Dec.	Y ← Y - 1, (Y) ← Rr	None	2	1
STD	Y+q,Rr	Store Indirect with Displacement	(Y + q) ← Rr	None	2	1
ST	Z,Rr	Store Indirect	(Z) ← Rr	None	2	1
ST	Z+,Rr	Store Indirect and Post-Inc.	(Z) ← Rr, Z ← Z + 1	None	2	1
ST	-Z,Rr	Store Indirect and Pre-Dec.	Z ← Z - 1, (Z) ← Rr	None	2	1
STD	Z+q,Rr	Store Indirect with Displacement	(Z + q) ← Rr	None	2	1
LPM		Load Program Memory	R0 ← (Z)	None	3	2
LPM	Rd,Z	Load Program Memory	Rd ← (Z)	None	3	2
LPM	Rd,Z+	Load Program Memory and Post-Inc	Rd ← (Z), Z ← Z+1	None	3	2
PUSH	Rr	Push Register on Stack	STACK ← Rr	None	2	1
POP	Rd	Pop Register from Stack	Rd ← STACK	None	2	1/2